

普通高等教育"十一五"国家级规划教材

高等院校信息安全专业系列教材

教育部高等学校信息安全专业教学指导委员会
中国计算机学会教育专业委员会
共同指导

顾问委员会主任：沈昌祥　编委会主任：肖国镇

信息安全数学基础
——算法、应用与实践

任　伟　编著

Information Security

http://www.tup.com.cn

根据教育部高等学校信息安全专业教学指导委员会编制的《高等学校信息安全专业指导性专业规范》组织编写

清华大学出版社

普通高等教育“十一五”国家级规划教材
高等院校信息安全专业系列教材

信息安全数学基础
——算法、应用与实践

任　伟　编著

清华大学出版社
北　京

内 容 简 介

本书包括初等数论、抽象代数、椭圆曲线论等方面的内容。本书选材合理、难度适中、层次分明、内容系统。书中以大量例题深入浅出地阐述信息安全数学基础各分支的基本概念、基本理论与基本方法，注重将抽象的理论与算法和实践相结合，并强调理论在信息安全特别是密码学中的具体应用实例。本书语言通俗易懂，容易自学。

本书可作为高等院校信息安全、计算机科学与技术、密码学、通信工程、信息对抗、电子工程等领域的研究生和本科生相关课程的教科书，也可作为这些领域的教学、科研和工程技术人员的参考书。

图书在版编目（CIP）数据

信息安全数学基础：算法、应用与实践/任伟编著. —北京：清华大学出版社，2016
高等院校信息安全专业系列教材
ISBN 978-7-302-41637-1

Ⅰ. ①信…　Ⅱ. ①任…　Ⅲ. ①信息安全－应用数学－高等学校－教材　Ⅳ. ①TP309 ②029

中国版本图书馆 CIP 数据核字(2015)第 228404 号

责任编辑： 张　民　李　晔
封面设计： 何凤霞
责任校对： 梁　毅
责任印制： 宋　林

出版发行： 清华大学出版社
网　　址： http://www.tup.com.cn，http://www.wqbook.com
地　　址： 北京清华大学学研大厦 A 座　　**邮　　编：** 100084
社 总 机： 010-62770175　　**邮　　购：** 010-62786544
投稿与读者服务： 010-62776969，c-service@tup.tsinghua.edu.cn
质量反馈： 010-62772015，zhiliang@tup.tsinghua.edu.cn
课件下载： http://www.tup.com.cn，010-62795954
印 装 者： 三河市中晟雅豪印务有限公司
经　　销： 全国新华书店
开　　本： 185mm×260mm　　**印　　张：** 8.75　　**字　　数：** 214 千字
版　　次： 2016 年 1 月第 1 版　　**印　　次：** 2016 年 1 月第1 次印刷
印　　数： 1～2000
定　　价： 25.00 元

产品编号：046936-01

高等院校信息安全专业系列教材

编审委员会

出版说明

21 世纪是信息时代，信息已成为社会发展的重要战略资源，社会的信息化已成为当今世界发展的潮流和核心，而信息安全在信息社会中将扮演极为重要的角色，它会直接关系到国家安全、企业经营和人们的日常生活。随着信息安全产业的快速发展，全球对信息安全人才的需求量不断增加，但我国目前信息安全人才极度匮乏，远远不能满足金融、商业、公安、军事和政府等部门的需求。要解决供需矛盾，必须加快信息安全人才的培养，以满足社会对信息安全人才的需求。为此，教育部继 2001 年批准在武汉大学开设信息安全本科专业之后，又批准了多所高等院校设立信息安全本科专业，而且许多高校和科研院所已设立了信息安全方向的具有硕士和博士学位授予权的学科点。

信息安全是计算机、通信、物理、数学等领域的交叉学科，对于这一新兴学科的培养模式和课程设置，各高校普遍缺乏经验，因此中国计算机学会教育专业委员会和清华大学出版社联合主办了“信息安全专业教育教学研讨会”等一系列研讨活动，并成立了“高等院校信息安全专业系列教材”编审委员会，由我国信息安全领域著名专家肖国镇教授担任编委会主任，指导“高等院校信息安全专业系列教材”的编写工作。编委会本着研究先行的指导原则，认真研讨国内外高等院校信息安全专业的教学体系和课程设置，进行了大量前瞻性的研究工作，而且这种研究工作将随着我国信息安全专业的发展不断深入。经过编委会全体委员及相关专家的推荐和审定，确定了本丛书首批教材的作者，这些作者绝大多数都是既在本专业领域有深厚的学术造诣、又在教学第一线有丰富的教学经验的学者、专家。

本系列教材是我国第一套专门针对信息安全专业的教材，其特点是：

① 体系完整、结构合理、内容先进。

② 适应面广：能够满足信息安全、计算机、通信工程等相关专业对信息安全领域课程的教材要求。

③ 立体配套：除主教材外，还配有多媒体电子教案、习题与实验指导等。

④ 版本更新及时，紧跟科学技术的新发展。

为了保证出版质量，我们坚持宁缺毋滥的原则，成熟一本，出版一本，并保持不断更新，力求将我国信息安全领域教育、科研的最新成果和成熟经验反映到教材中来。在全力做好本版教材，满足学生用书的基础上，还经由专

家的推荐和审定，遴选了一批国外信息安全领域优秀的教材加入到本系列教材中，以进一步满足大家对外版书的需求。热切期望广大教师和科研工作者加入我们的队伍，同时也欢迎广大读者对本系列教材提出宝贵意见，以便我们对本系列教材的组织、编写与出版工作不断改进，为我国信息安全专业的教材建设与人才培养做出更大的贡献。

“高等院校信息安全专业系列教材”已于 2006 年年初正式列入普通高等教育“十一五”国家级教材规划(见教高[2006]9 号文件《教育部关于印发普通高等教育“十一五”国家级教材规划选题的通知》)。我们会严把出版环节，保证规划教材的编校和印刷质量，按时完成出版任务。

2007 年 6 月，教育部高等学校信息安全类专业教学指导委员会成立大会暨第一次会议在北京胜利召开。本次会议由教育部高等学校信息安全类专业教学指导委员会主任单位北京工业大学和北京电子科技学院主办，清华大学出版社协办。教育部高等学校信息安全类专业教学指导委员会的成立对我国信息安全专业的发展起到重要的指导和推动作用。2006 年教育部给武汉大学下达了“信息安全专业指导性专业规范研制”的教学科研项目。2007 年起该项目由教育部高等学校信息安全类专业教学指导委员会组织实施。在高教司和教指委的指导下，项目组团结一致，努力工作，克服困难，历时 5 年，制定出我国第一个信息安全专业指导性专业规范，于 2012 年年底通过经教育部高等教育司理工科教育处授权组织的专家组评审，并且已经得到武汉大学等许多高校的实际使用。2013 年，新一届“教育部高等学校信息安全专业教学指导委员会”成立。经组织审查和研究决定，2014 年以“教育部高等学校信息安全专业教学指导委员会”的名义正式发布《高等学校信息安全专业指导性专业规范》(由清华大学出版社正式出版)。“高等院校信息安全专业系列教材”在教育部高等学校信息安全专业教学指导委员会的指导下，根据《高等学校信息安全专业指导性专业规范》组织编写和修订，进一步体现科学性、系统性和新颖性，及时反映教学改革和课程建设的新成果，并随着我国信息安全学科的发展不断完善。

我们的 E-mail 地址：zhangm@tup.tsinghua.edu.cn；联系人：张民。

“高等院校信息安全专业系列教材”编审委员会

前 言

随着中央网络安全与信息化领导小组的成立，信息安全进入公众的视野，它不仅关系到国防军事等重大战略问题以及国计民生等新兴战略产业的发展，而且与每个人的日常生活息息相关。目前，我国信息安全所面临的形势十分严峻，信息安全学科的发展已经刻不容缓，信息安全学科升级为国家一级学科也已经提上议事日程。

信息安全数学是信息安全学科的理论基础，其内容涉及面较广，例如数论与有限域等在信息安全的重要基础课如密码学中有大量的应用。信息安全数学基础是信息安全专业一门重要的基础必修课程。此外，信息安全数学在计算机科学、信息与通信工程、网络工程、电子对抗等学科中也都有着重要的应用。

目前信息安全数学方面的书籍有些难以读懂，这在一定程度上阻碍了信息安全学科以及信息安全知识的普及。对抽象的数学知识介绍较多，虽然一定程度上可以锻炼学生的抽象思维能力，但容易造成学生对所学内容的畏难情绪。另外，单纯的理论知识介绍会导致学生不清楚这些理论如何应用，从而对所学内容不能留下较深刻的印象。一些来自计算机科学、通信工程、网络工程等专业的学生虽然对信息安全方向感兴趣，但是因为信息安全数学知识的抽象和难以普及导致无法将本专业与信息安全方向结合起来。

本书重点强调信息安全数学基础在信息安全中的应用，并通过实践（算法与编程）环节强化对理论的理解。减少了一些在信息安全中应用较少的非重点数学理论，注重从计算机科学（算法）角度介绍而不是从纯数学角度介绍。强调抽象知识的算法解释和形象化，便于读者自学和易于教学。

本书在写作过程中特别遵循了以下思路。

（1）体例新颖活泼、语言通俗易懂、精心安排示例。注意到目前市场上“大话×××”、“×××趣谈”、“图解×××”等图书深受读者喜爱，本书在保证论述的严谨性前提下，语言尽量形象生动、文风尽量活泼，以激发学习者的兴趣。根据作者对“信息安全数学基础”这一课程多年的教学实践经验，给出一些较为独特的比喻，虽然有些比较浅显，但主要目的是让读者特别是初学者快速理解，印象深刻，阅读轻松。

（2）内容编排独特、循序渐进、由浅入深。注重内容之间的联系和讲解先后次序。内容选取尽量考虑到重要性和必要性。注重给出一些浅显易懂的类比，便于读者建立所学知识与前后内容之间的联系。

(3) 以应用为导向，理论联系实际。不单纯讲解数学基础，而是从应用需要的角度出发，着重讲解基础知识点和关键点，突出实用性和可操作性。注重对算法和实践能力的培养，书中重点介绍计算数论(算法数论)中的算法，鼓励读者自主实现这些算法来提高实践能力。

(4) 注重启发性和对创新能力的培养。通过在正文中设立“思考”环节，以提高启发性并激发读者思考。在内容组织中潜移默化地强调数学素养的培养，根据数学内容的需要，采用合情归纳法、演绎法、公理集合论方法等多种论述方法。

全书共分 12 章：第 1 章整除；第 2 章同余；第 3 章同余式；第 4 章二次同余式和平方剩余；第 5 章原根与指数；第 6 章群；第 7 章环与域；第 8 章素性检测；第 9 章椭圆曲线群；第 10 章大整数分解算法；第 11 章离散对数算法；第 12 章其他高级应用。其中，第 9～12 章为高级部分，高级部分与部分打星号的章节可选学。全书授课学时为 40～64 学时。

本书面向的主要对象包括从事信息和网络安全研究的科研人员，学习信息安全相关课程的高等院校信息安全类、计算机科学类、信息与计算科学类专业本科生，以及从事信息安全技术研发、应用和管理的工程技术人员。

本书受到了国家自然科学基金面上项目(No. 61170217)，以及湖北省教育厅高等学校教学研究项目(No. 2015A06)的支持，在此表示感谢。感谢研究生叶敏、刘宇靓、林佳华、曹强、曾玲玲的辅助性工作。

愿本书的写作能为我国信息安全数学的教学和普及起到一点抛砖引玉的作用。由于作者水平和学识有限，不足之处在所难免，在此衷心恳请广大读者、同行批评指正。联系方式是 weirencs@cug. edu. cn。

作　者

目 录

基 础 篇

高 级 篇

基　础　篇

第1章 整　除

在整数集合中，整除是一种重要的二元关系，相关概念和性质包括素数、公因数、欧几里得除法（辗转相除法，Euclid 除法）、算术基本定理等。这些概念和性质又是整数集合中另一种重要的二元关系——同余关系的基础。本章先介绍整除，第 2 章介绍同余。

本章的重点是 Euclid 除法和 Euclid 算法，难点是扩展的 Euclid 算法。

1.1 整除的概念

通常，用 Z 表示整数集合，整数即为 $0,\pm1,\pm2,\cdots$。

自然数是非负整数，用 N 来表示。

定义 1.1（整除）　设 a,b 是任意两个整数，其中 $b\neq0$。如果存在一个整数 q，使得等式

$$a = qb$$

成立，则称 b 整除 a 或者 a 被 b 整除，记作 $b|a$。b 叫做 a 的因数，a 叫做 b 的倍数。q 写成 a/b 或者 $\frac{a}{b}$。否则，称 b 不能整除 a，或者 a 不能被 b 整除，记作 $b \quad a$。

注意，这里整除的定义通过乘法运算给出的（而不是通过除法运算定义的）；通过整数 q 的存在性表述整除性。另外，符号 $b|a$ 本身就包含了 $b\neq0$。

例 1.1　请写出 20 的所有因数。

解答：$\pm1,\pm2,\pm4,\pm5,\pm10,\pm20$。

根据定义，有：

0 是任何非零整数的倍数，即 $a|0$，这里 $a\neq0,a\in Z$；

1 是任何整数的因数，即 $1|a,a\in Z$；

任何非零整数 a 是自己的倍数，也是自己的因数，即 $a|a$，这里 $a\neq0,a\in Z$。

整除有如下性质：

例 1.2　设 a,b 为整数。若 $b|a$，则 $b|(-a),(-b)|a,(-b)|(-a),|b|\,|\,|a|$。

证明：由 $b|a$，于是存在整数 q，使得 $a=qb$。

要证明所需结论，即需要证明存在整数 Q，使得等式 $(-a)=Qb,a=Q(-b),(-a)=Q(-b),|a|=Q|b|$ 成立。

由条件 $a=qb$ 通过简单的推理可以发现，当 Q 分别为 $-q,-q,q,|q|$ 时，上述等式满足。于是可知，相应的整数 Q 存在。■

由这个例子可知，可将重点放在正整数的整除上来。

上述证明的思路在于：从已知条件和证明目标同时入手，变换转换，中间相遇。具体而言，由整除的概念得到相应等式，由相应等式推出整数 Q 的存在性，由整数 Q 的存在性推出整除性。

由这一思路，可以证明整除的如下性质（请读者自行给出证明并给出实例）：

定理 1.1（传递性） 设 $a\neq0,b\neq0,c$ 是三个整数。若 $a|b,b|c$，则 $a|c$。

定理 1.2 设 $a,b,c\neq0$ 是三个整数。若 $c|a,c|b$，则 $c|a\pm b$。

定理 1.3 设 $a,b,c\neq0$ 是三个整数。若 $c|a,c|b$，则对任意整数 s,t，有

$$c|sa\pm tb$$

（提示：Q 分别为 $q_1q_2,q_1\pm q_2,sq_1\pm tq_2$）

例 1.3 设 $a,b,c\neq0$ 是三个整数，对于 $c|a,c|b$，如果存在整数 s,t，使得 $sa+tb=1$，则 $c=\pm1$。

证明：因为 $c|a,c|b$，存在整数 s,t，使得 $sa+tb=1$，于是由定理 1.3，有 $c|sa+tb=1$，于是，$c=\pm1$。 ■

由整除和因数的概念，可以根据因数情况对整数进行分类。

定义 1.2 设整数 $n\neq0,\pm1$，如果除了平凡因数 $\pm1,\pm n$ 外，n 没有其他因数，那么 n 叫做**素数**（或**质数**、**不可约数**），否则 n 叫做**合数**。

当整数 $n\neq0,\pm1$ 时，n 和 $-n$ 同为素数或合数。因此，若没有特别声明，素数总是指正整数，通常写成 p。

思考 1.1：请写出 30 以内的素数。

（答案：2，3，5，7，11，13，17，19，23，29）

下面证明每个合数必有素因数。

定理 1.4 设 n 是一个正合数，p 是 n 的一个大于 1 的最小正因数，则 p 一定是素数，且 $p\leqslant\sqrt{n}$。

证明：反证法。若 p 不是素数，则存在整数 $q,1<q<p$，使得 $q|p$，由条件知 $p|n$，于是根据定理 1.1，有 $q|n$，这与 p 是 n 的大于 1 的最小正因数矛盾。所以 p 是素数。

若 $p>\sqrt{n}$ 成立，则 n 的另一个因数 $n/p<n/\sqrt{n}=\sqrt{n}$，于是，n/p 是一个比 p 小的因数，这与 p 是 n 的大于 1 的最小正因数矛盾。证毕。 ■

非正式地说，上述定理说明了两点：素因数可以视为合数的“组成成分”。且这一“组成成分”中必然有一个小于等于 $\sqrt{n}$。

定理 1.4 给出了寻找素数的有效方法。为了求出不超过给定正整数 $x(x>1)$ 的所有素数，只要把从 2 到 x 的所有合数都删去即可。因为不超过 x 的合数 n 必有一个素因子 $p\leqslant\sqrt{n}\leqslant\sqrt{x}$，所以只要先求出 $\sqrt{x}$ 以内的全部素数 $\{p_i,1\leqslant i\leqslant k\}$（其中，$k$ 为 $\sqrt{x}$ 以内的素数个数），然后把不超过 x 的 p_i 的倍数（p_i 本身除外）全部删去，剩下的就正好是不超过 x 的全部素数。这种寻找素数的方法称为 Eratosthenes 筛法。

例 1.4 求出不超过 64 的所有素数。

解答：先求出不超过 $\sqrt{64}=8$ 的所有素数，依次为 2，3，5，7，然后从 2～64 的所有整

数依次删去除了 2、3、5、7 以外的 2 的倍数、3 的倍数、5 的倍数、7 的倍数，剩下的即为所求。具体过程如下所示：

2	3	~~4~~	5	~~6~~	7	~~8~~	~~9~~	~~10~~	11	~~12~~	13	~~14~~	~~15~~
~~16~~	17	~~18~~	19	~~20~~	~~21~~	~~22~~	23	~~24~~	~~25~~	~~26~~	~~27~~		
~~28~~	29	~~30~~	31	~~32~~	~~33~~	~~34~~	~~35~~	~~36~~	37	~~38~~	~~39~~		
~~40~~	41	~~42~~	43	~~44~~	~~45~~	~~46~~	47	~~48~~	49	~~50~~	51		
~~52~~	53	~~54~~	~~55~~	~~56~~	~~57~~	~~58~~	59	~~60~~	61	~~62~~	~~63~~	~~64~~	

可见，没有删去的数是：2、3、5、7、11、13、17、19、23、29、31、37、41、43、47、51、53、59、61。这些即为不超过 64 的所有素数。

依据上述方法可以编写一个算法，输出不超过输入值的所有素数。

一个很自然会想到的问题就是：素数是否可以穷举？下面的证明说明素数的数量有无穷多个。

定理 1.5　素数有无穷多个。

证明：反证法。假设有有限个素数，则不妨设它们为 $p_1, p_2, \cdots, p_n$。考虑大于 1 的整数

$$N = p_1 p_2 \cdots p_n + 1$$

容易看到：p_1，p_2，…，p_n 都不能整除 N，于是 N 没有素因数，由定理 1.4 知，N 不是合数，于是 N 为素数。这与有限个素数矛盾。■

公元前 3 世纪古希腊大数学家欧几里得(Euclid)在 *The Elements*（中文译名为《几何原本》）一书中给出了该证明方法，成了一种经典的“构造矛盾”的反证法。[①]

可以看到，本节论述的过程遵循了数学公理化方法，该方法从基本概念和公理出发，通过证明逐步扩充定理和性质。

1.2　Euclid 算法

前面讨论的是整除的情况，如果不能整除时会如何呢？

定理 1.6(Euclid 除法，也称为带余除法)　设 a, b 是两个整数，其中 $b>0$，则存在唯一的整数 q, r 使得

$$a = qb + r, \quad 0 \leqslant r < b \tag{1.1}$$

证明：先证明存在性。考虑一个整数序列

$$\cdots, -3b, -2b, -b, 0, b, 2b, 3b, \cdots$$

它们将实数轴分成长度为 b 的一系列区间，而 a 必定落在其中的一个区间上。因此存在一个整数 q，使得

$$qb \leqslant a < (q+1)b$$

① 该命题为《几何原本》第 9 卷第 20 个命题，编号为 IX. 20，原命题是：预先给定几个质数，那么有比它们更多的质数。该证明被《来自天书的证明》一书收录为数学史上的经典证明。(类似的方法包括 Cantor 的对角线反证法，Turing 的停机问题的不可判定性，以及 Godel 的不完备性定理。)欧几里得的《几何原本》是西方数学公理化方法的起源和数学逻辑演绎推导的代表。《九章算术》为代表的中国数学则是以归纳计算和构造为主要方法。

令 $r=a-qb$，则有

$$a = qb+r, 0\leqslant r<b$$

再证明唯一性。如果分别有 q_1, r_1 和 q_2, r_2 满足(1.1)式，则

$$a = q_1 b + r_1,\quad 0\leqslant r_1<b$$
$$a = q_2 b + r_2,\quad 0\leqslant r_2<b$$

两式相减，有

$$(q_1-q_2)b=-(r_1-r_2)$$

当 $q_1\neq q_2$ 时，左边的绝对值$\geqslant b$，而右边的绝对值$<b$，这是不可能的。于是，$q_1=q_2$，$r_1=r_2$。证毕。■

定义 1.3 (1.1)式中的 q 叫做 a 被 b 除所得的不完全商，r 叫做 a 被 b 除所得的余数。

Euclid 除法可以理解成用一个长度为 b 的“尺子”去度量长度 a，度量最后剩下的一段 r 不会大于“尺子”的长度 b。

Euclid 除法的用途是可以将两个数之间整除关系的判定问题转化为计算问题。判断 a 是否能被非零整数 b 整除的充要条件是 a 被 b 除所得的余数 $r=0$。

通常，$0\leqslant r<b$，这时 r 叫做最小非负余数；但在有些时候通过“平移”(即调整不完全商的大小，一般是加 1)，可以将 r 调整为 $|r|\leqslant b/2$，这时 r 叫做绝对值最小余数，它在后面介绍的 Euclid 算法中(见例 1.8)能起到算法加速的作用。

思考 1.2 令 $b=7$，则最小非负余数为多少？绝对值最小余数为多少？

解答：$r=0,1,2,3,4,5,6$ 为最小非负余数；

$r=-3,-2,-1,0,1,2,3$ 为绝对值最小余数。

定义 1.4(公因子) 设 $a_1,\cdots,a_n$ 是 $n(n\geqslant 2)$个整数。若整数 d 是它们中每一个数的因数，那么 d 就叫做 $a_1,\cdots,a_n$ 的一个公约数(也叫公因数)。

d 是 $a_1,\cdots,a_n$ 的一个公因数的数学表达式为：

$$d\mid a_1,\cdots,d\mid a_n$$

如果整数 $a_1,\cdots,a_n$ 不全为零，那么 $a_1,\cdots,a_n$ 的所有公约数中最大的一个公约数叫做最大公约数(Greatest Common Divisor)，记作 $\gcd(a_1,\cdots,a_n)$或$(a_1,\cdots,a_n)$。

特别地，当$(a_1,\cdots,a_n)=1$，称$(a_1,\cdots,a_n)$互素或互质。

最大公约数的等价定义为：

$d>0$ 是 $a_1,\cdots,a_n$ 的最大公约数的数学表达式可以表述为：

(1) $d\mid a_1,\cdots,d\mid a_n$。

(2) 若 $e\mid a_1,\cdots,e\mid a_n$，则 $e\mid d$。

条件(1)说明 d 是公约数；条件(2)说明 d 在公约数中最大。

下面的定理给出了最大公约数的另一个等价定义：

定理 1.7 a,b 是不全为零的整数，a,b 的最大公约数 $d=(a,b)$是集合

$$\{sa+tb \mid s, t\in Z\}$$

中的最小正整数。

证明：令集合$\{sa+tb\mid s,t\in Z\}$中最小的正整数为 m。下面证明 $m=d$，方法是先证明

$d|m$，再证明 $m|d$。

一方面，因为 $d=(a,b)$，于是 $d|a,d|b$，由定理 1.3，$d|sa+tb,s,t\in Z$，于是 $d|m$。

另一方面，由带余除法知，存在整数 q_1,q_2,r_1,r_2，使得

$$a=q_1m+r_1,\ b=q_2m+r_2,\text{其中 } 0\leqslant r_1,r_2<m$$

易知，$r_1,r_2\in\{sa+tb|s,t\in Z\}$，由于 m 是该集合中的最小正整数，故 $r_1=r_2=0$，即 $m|a,m|b$，由最大公约数的定义知，$m|d$。

综合 $d|m,m|d$，且均为正整数，有 $m=d$。证毕。■

定理 1.7 从线性组合的角度来考查最大公因子。如果从几何的角度来观察，这可以视为 $y=x$ 斜线与格子割线中那个最短的线段。例如图 1.1 中斜线下方 a 上方第一个线段对应的值为 $a-2b$，斜线下方 $2a$ 上方线段对应的值为 $2a-4b$。因此，(a,b) 也可以视为利用长度为 a 和 b 的两把“尺子”可以“丈量”的最小长度。

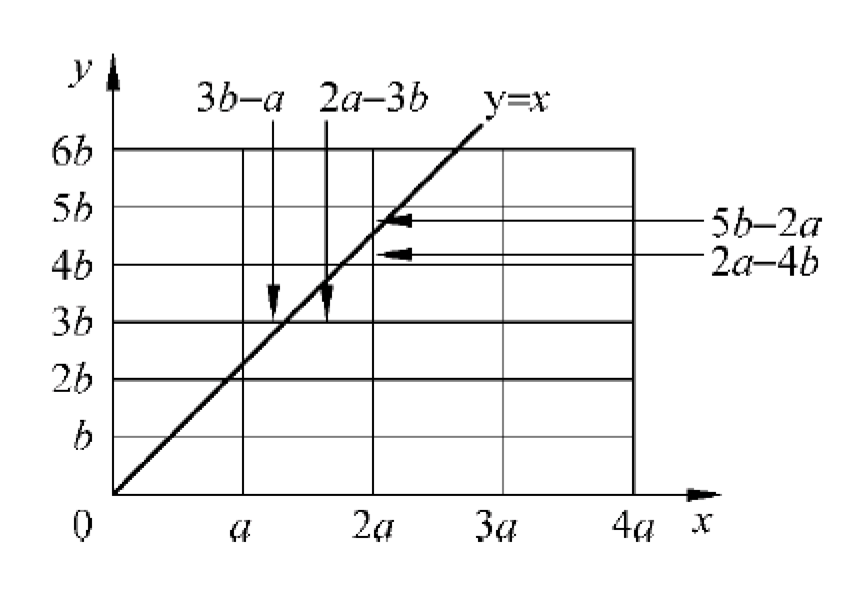


图 1.1　a,b 的线性组合

推论 1.1　a,b 是不全为零的整数，a,b 的最大公约数 $d=(a,b)$，集合

$$\{sa+tb|s,t\in Z\}$$

由 d 的所有倍数组成。

这再一次说明了 d 是长度为 a 和 b 的“尺子”组合起来可以“丈量”的“最小长度单位”。有了这个“单位长度”，则可以“反复”丈量出其他长度。

定理 1.8　设 a,b 为不全为零的整数，则

方程 $ax+by=c$ 有整数解，当且仅当 $c\in\{ax+by|x,y\in Z\}$，即当且仅当 $(a,b)|c$。

该定理可用于判定二元一次不定方程 $ax+by=c(a,b,c\in Z)$ 的整数解是否存在。

特别地，若 $(a,b)=1$，则存在整数 x,y，使得 $ax+by=1$。

这一特例也说明只有当 a 和 b 互素时，$ax\equiv 1 \pmod b$ 才有解。

定理 1.9　设 $a_1,\cdots,a_n$ 是 n 个不全为零的整数，则

(1) $a_1,\cdots,a_n$ 与 $|a_1|,\cdots,|a_n|$ 的公因数相同。

(2) $(a_1,\cdots,a_n)=(|a_1|,\cdots,|a_n|)$。

这个定理将公因数的讨论转化为在非负数范围内讨论。

定理 1.10　设 b 是任一正整数，则 $(0,b)=b$。

平凡的求两个整数最大公因子的方法是分别求出两个数的因子，然后挑出它们中最大的公因子。这种方法在两个数比较小的情况下是可行的，但是当两个数比较大时，分解其因子是十分困难的，而且这个方法的效率不高，因此，求两个整数的最大公因子需要寻找更好的办法。

定理 1.11 设 a,b,r 是三个不全为零的整数。如果

$$a=qb+r$$

其中 q 是整数，则 $(a,b)=(b,r)$

证明： 令 $d=(a,b)$，$d'=(b,r)$，$d|a$，$d|b$，由定理 1.3，

$$d|a+(-q)b=r$$

于是 d 是 b,r 的公因数，从而 $d\leqslant d'$。

同理，d' 是 a,b 的公因数，从而 $d'\leqslant d$。

因此，$d=d'$。 ■

例 1.5 因为 $1573=5\cdot 286+143$，所以 $(1573,286)=(286,143)=143$

以上定理的用处是：可以将求两个较大数的公因数转化为求两个较小数的公因数。利用这一思想，可以得到计算求最大公因数的方法。这一算法称为“欧几里得除法”，也称为“辗转相除法”。它可能是世界上最著名和最早的算法。

从算法递归调用的角度描述来定理 1.12，就是 $\gcd(a,b)=\gcd(b,\ a \bmod b)$。

算法 1.1 Euclid 算法递归形式：计算两个整数的最大公因子。

输入：两个非负整数 a,b，且 $a\geqslant b$。（先将待计算的整数取绝对值）

输出：a,b 的最大公因子。

```
GCD(a,b) {
If b<>0
    Return GCD(b, a mod b);
Else
    Return a;
}
```

递归形式便于理解，但不便于了解算法的实际过程。下面给出非递归形式的 Euclid 算法。在给出非递归形式之前，先看一个例子。

例 1.6 计算 $\gcd(4864,3458)=38$ 的分解步骤。

解答：

$$\begin{aligned}
4864 &= 1\cdot 3458+1406\\
3458 &= 2\cdot 1406+646\\
1406 &= 2\cdot 646+114\\
646 &= 5\cdot 114+76\\
114 &= 1\cdot 76+38\\
76 &= 2\cdot 38+0
\end{aligned}$$

算法 1.2 Euclid 算法：计算两个整数的最大公因子。

输入：两个非负整数 a,b，且 $a\geqslant b$。（先将待计算的整数取绝对值）

输出：a,b 的最大公因子。

```
GCD(a,b) {
    While (b≠0) do
    r←a mod b
    a←b
    b←r
    Return a
}
```

思考 1.3：算法 1.2 有可能是死循环吗？

算法 1.2 中的循环会结束（不会是死循环），因为 r 逐步在减小，最终变为 0（从而退出循环）。

例 1.7 用算法 1.2 计算 gcd(169,121)=1 的分解步骤。

解答：

$$169 = 1 \cdot 121 + 48$$
$$121 = 2 \cdot 48 + 25$$
$$48 = 1 \cdot 25 + 23$$
$$25 = 1 \cdot 23 + 2$$
$$23 = 11 \cdot 2 + 1$$
$$2 = 2 \cdot 1 + 0$$

如果利用绝对值最小余数代替最小非负余数，可以对算法 1.2 进行优化。请见下面的例子。

例 1.8 设 $a=46480, b=39423$，计算 (a,b)。

解答：利用 Euclid 除法。

方法 1：使用最小非负余数。

$$46480=1 \cdot 39423+7057$$
$$39423 = 5 \cdot 7057+4138$$
$$7057=1 \cdot 4138+2919$$
$$4138=1 \cdot 2919+1219$$
$$2919=2 \cdot 1219+481$$
$$1219=2 \cdot 481+257$$
$$481 = 1 \cdot 257+224$$
$$257 = 1 \cdot 224+33$$
$$224=6 \cdot 33+26$$
$$33=1 \cdot 26+7$$
$$26=3 \cdot 7+5$$
$$7=1 \cdot 5+2$$
$$5=2 \cdot 2+1$$
$$2=2 \cdot 1+0$$

方法 2：使用绝对值最小余数。

$$46480=1\cdot 39423+7057$$
$$39423=6\cdot 7057-2919$$
$$7057=2\cdot 2919+1219$$
$$2919=2\cdot 1219+481$$
$$1219=3\cdot 481-224$$
$$481=2\cdot 224+33$$
$$224=7\cdot 33-7$$
$$33=5\cdot 7-2$$
$$7=3\cdot 2+1$$
$$2=2\cdot 1+0$$

所以(46480,39423)＝1。

容易看到，方法 2 要比方法 1 运算次数少一些，所以绝对值最小余数方法可以加快计算，减少计算时间。

1.3 扩展的 Euclid 算法

下面的推导给出了 Euclid 算法的一个严格证明。

设 a,b 是任意两个正整数。记 $r_{-2}=a,r_{-1}=b$，反复运用带余除法，有：

$$\begin{aligned}
r_{-2} &= q_0 r_{-1} + r_0 \quad 0\leqslant r_0 < r_{-1}\\
r_{-1} &= q_1 r_0 + r_1 \quad 0\leqslant r_1 < r_0\\
r_0 &= q_2 r_1 + r_2 \quad 0\leqslant r_2 < r_1\\
&\cdots\\
r_{n-3} &= q_{n-1} r_{n-2} + r_{n-1} \quad 0\leqslant r_{n-1} < r_{n-2}\\
r_{n-2} &= q_n r_{n-1} + r_n \quad 0\leqslant r_n < r_{n-1}\\
r_{n-1} &= q_{n+1} r_n + r_{n+1} \quad r_{n+1} = 0
\end{aligned} \tag{1.2}$$

容易看到，上述过程为从数列 $r_{-2}, r_{-1}, r_0, r_1, \cdots, r_n$ 从前两项依次求出第三项，直到最后一项的过程。有

$$(a,b)=(r_{-2},r_{-1})=(r_{-1},r_0)=\cdots=(r_{n-1},r_n)=(r_n,r_{n+1})=(r_n,0)=r_n$$

如果将这一过程反方向写出，有：

$$\begin{aligned}
r_n &= r_{n-2} - q_n r_{n-1}\\
r_{n-1} &= r_{n-3} - q_{n-1} r_{n-2}\\
&\cdots\\
r_2 &= r_0 - q_2 r_1\\
r_1 &= r_{-1} - q_1 r_0\\
r_0 &= r_{-2} - q_0 r_{-1}
\end{aligned}$$

即，r_n，r_{n-1}，…，r_1，r_0中每一项均可以用后两项表示。于是 r_n可以用r_{-2}，r_{-1}表示，即可以找到整数 s，t，使得

$$sa + tb = r_n = (a,b)$$

例 1.9　$a=169$，$b=121$，求整数 s，t，使得 $sa + tb = (a,b)$。

解答：将例 1.7 的过程反过来写，有

$$\begin{aligned} 1 &= 23 - 11 \cdot 2 \\ &= 23 - 11 \cdot (25 - 1 \cdot 23) \\ &= -11 \cdot 25 + 12 \cdot (48 - 1 \cdot 25) \\ &= 12 \cdot 48 - 23 \cdot (121 - 2 \cdot 48) \\ &= -23 \cdot 121 + 58 \cdot (169 - 1 \cdot 121) \\ &= 58 \cdot 169 - 81 \cdot 121 \end{aligned}$$

因此，整数 $s=58$，$t=-81$，满足 $sa+tb=(a,b)$。

下面推导一个算法，用于在给定 a 和 b 情况下计算 s 和 t，使得 $sa+tb=(a,b)$。该算法具有重要的应用，如当 a，b 互素时，可求出 s 来，满足 $sa + tb = 1$，即 $sa = 1 \pmod b$。(第 6 章将会介绍，s 在 Z_b^* 的乘法群中称为 a 的乘法逆元。)这一算法通常称为“扩展的欧几里得”(Extended Euclid)算法。

在给出具体算法之前，首先观察和分析计算中存在的递推关系。

定理 1.12　设 a，b 是任意两个正整数，则

$$s_n a + t_n b = (a,b) \tag{1.3}$$

对于 $j=0,1,\cdots,n-1$，这里 s_j，t_j归纳地定义为

$$\begin{aligned} &s_{-2} = 1,\ s_{-1} = 0,\ s_j = s_{j-2} - q_j s_{j-1} \\ &t_{-2} = 0,\ t_{-1} = 1,\ t_j = t_{j-2} - q_j t_{j-1},\quad j = 0,1,2,\cdots,n-1,n \end{aligned} \tag{1.4}$$

其中 $q_j=[r_{j-2}/r_{j-1}]$是(1.2)式中的不完全商。

(设 $x\in R$，$[x]$表示不超过 x 的最大整数，称作实数 x 的整数部分。)

证明　只需证明：对于 $j=-2,-1,0,1,\cdots,n-1$

$$s_j a + t_j b = r_j \tag{1.5}$$

其中 $r_j=r_{j-2}-q_j r_{j-1}$是(1.2)式中的余数。因为$(a,b)=r_n$，所以

$$s_n a + t_n b = (a,b)$$

对 j 用数学归纳法来证明(1.5)式。

$j=-2$ 时，有 $s_{-2}=1$，$t_{-2}=0$ 以及

$$s_{-2}a + t_{-2}b=a=r_{-2}$$

(1.5)式对于 $j=-2$ 成立。

$j=-1$ 时，有 $s_{-1}=0$，$t_{-1}=1$ 以及

$$s_{-1}a + t_{-1}b=b=r_{-1}$$

(1.5)式对于 $j=-1$ 成立。

假设(1.5)式对于 $-2 \leqslant j \leqslant k-1$ 成立，即

$$s_j a + t_j b = r_j$$

对于 $j=k$，有

$$r_k = r_{k-2} - q_k r_{k-1},$$

利用归纳假设，得到

$$\begin{aligned} r_k &= (s_{k-2}a + t_{k-2}b) - q_k(s_{k-1}a + t_{k-1}b) \\ &= (s_{k-1} - q_k s_{k-1})a + (t_{k-2} - q_k t_{k-1})b \\ &= s_k a + t_k b \end{aligned}$$

因此，(1.5)式对于 $j = k$ 也成立。

根据数学归纳法，(1.5)式对所有的 $j = -2, -1, 0, 1, \cdots, n-1$ 成立。 ■

有了关于 s 的递推关系 $s_j = s_{j-2} - q_j s_{j-1}$ 以及关于 t 的递推关系 $t_j = t_{j-2} - q_j t_{j-1}$，便可以求出最终需要的 s 和 t，即 s_n 和 t_n。

下面根据定理 1.13 设计一个算法：假设 a 和 b 是不全为零的非负整数，计算 s, t 使得 $sa + tb = (a, b)$。

首先，令

$$r_{-2} = a, \quad r_{-1} = b$$
$$s_{-2} = 1, \quad s_{-1} = 0$$
$$t_{-2} = 0, \quad t_{-1} = 1$$

(1) 如果 $r_{-1} = 0$，则令

$$s = s_{-2}, \quad t = t_{-2}$$

否则，计算

$$q_0 = [r_{-2}/r_{-1}], \quad r_0 = r_{-2} - q_0 r_{-1}$$

(2) 如果 $r_0 = 0$，则令

$$s = s_{-1}, \quad t = t_{-1}$$

否则，计算

$$s_0 = s_{-2} - q_0 s_{-1}, \quad t_0 = t_{-2} - q_0 t_{-1}$$

以及

$$q_1 = [r_{-1}/r_0], \quad r_1 = r_{-1} - q_1 r_0$$

(3) 如果 $r_1 = 0$，则令

$$s = s_0, \; t = t_0$$

否则，计算

$$s_1 = s_{-1} - q_1 s_0, \quad t_1 = t_{-1} - q_1 t_0$$

以及

$$q_2 = [r_0/r_1], \quad r_2 = r_0 - q_2 r_1$$

…

$(j+1)$若 $r_{j-1} = 0 (j \geqslant 3)$，则令

$$s = s_{j-2}, \quad t = t_{j-2}$$

否则，计算

$$s_{j-1} = s_{j-3} - q_{j-1} s_{j-2}, \quad t_{j-1} = t_{j-3} - q_{j-1} t_{j-2}$$

以及

$q_j = [r_{j-2}/r_{j-1}]$，$r_j = r_{j-2} - q_j r_{j-1}$

最后，一定有 $r_{n+1}=0$。这时，令

$$s=s_n,\quad t=t_n$$

总之，可以找到整数 s,t，使得

$$sa+tb=r_n=(a,b)$$

将上述推导过程写成如下算法：

算法 1.3　Extended Euclid 算法。

输入：两个非负整数，a,b，且 $a\geq b$。

输出：gcd(a,b)，以及满足 $sa+tb=\gcd(a,b)$ 的整数 s,t。

```
ExtendedEuclid(a,b){
(R,S,T)← (a,1,0);
(R',S',T')← (b,0,1);
While(R'≠0) do {
     q←[R/R'];
     (Temp1,Temp2,Temp3)← (R- qR',S- qS',T- qT')
     (R,S,T)← (R',S',T')
     (R',S',T')← (Temp1,Temp2,Temp3)
}
Return R,S,T;
}
```

例 1.10　写出 Extended Euclid 算法的计算过程，设 $a=1859$，$b=1573$，计算整数 s，t，使得 $sa+tb=(a,b)$。

解答：将算法写成表格形式

$R'\neq 0$?	q	R	R'	S	S'	T	T'
		a=1859	b=1573	1	0	0	1
Yes	1	1573	286	0	1	1	−1
Yes	5	286	143	1	−5	−1	6
Yes	2	143	⓪	−5	11	6	−13
No							

返回值

因此，$s=-5$，$t=6$，使得

$$(-5)\cdot 1859+6\cdot 1573=143$$

思考 1.4：算法 1.2 和 1.3 的区别。

算法 1.3 中引入两个关于 s 和 t 的递推关系。在计算 r 的递推关系求出 (a,b) 的同时，也计算了所需要的 s 和 t，因此，称为“扩展的”Euclid 算法。

值得强调的是，在实际中扩展的 Euclid 算法主要是用于求乘法逆元。例如，假设 $b<m$，当 $(m,b)=1$ 时，$sm+tb=(m,b)=1$，因此，$tb\equiv 1 \pmod m$。于是找到了 t，这是 b 在 Z_m^* 的乘法群中的乘法逆元(第 6 章将介绍群的概念)。为表达规整，算法中重新使用了不同的变量名和编排。见算法 1.4：

算法 1.4　Extended Euclid 算法求逆元。

输入：两个非负整数,m,b,且 $m>b$。

输出：b 在 $Z_m{}^*$ 中的乘法逆元。

```
ExtendedEuclid(m,b) {
1 (A1,A2,A3)← (1,0,m); (B1,B2,B3)← (0,1,b);
2 If B3=0 Return 'No Inverse';
3 If B3=1 Return B2;
4 Q←[A3/B3];
5 (Temp1,Temp2,Temp3)← (A1-QB1,A2-QB2,A3-QB3)
6 (A1,A2,A3)← (B1,B2,B3)
7 (B1,B2,B3)← (Temp1,Temp2,Temp3)
8 Goto 2
}
```

1.4 算术基本定理

在给出算术基本定理之前,简单介绍一下最大公约数和最小公倍数的性质。

定理 1.13　设 a,b 是任意两个不全为零的整数,

(1) 若 m 是任一正整数,则 $(am,bm)=(a,b)m$

(2) 若非零整数 d 满足 $d|a$,$d|b$,则 $(a/d,b/d)=(a,b)/d$。特别地,

$$(a/(a,b),b/(a,b))=1$$

定理 1.14　设 $a_1,\cdots,a_n$ 是 n 个整数,且 $a_1\neq 0$。令

$$(a_1,a_2)=d_2,(d_2,a_3)=d_3,\cdots,(d_{n-1},a_n)=d_n$$

则 $(a_1,\cdots,a_n)=d_n$。

该定理说明多个整数的公约数可以通过逐个计算得到。

定理 1.15　设 a、b、c 是三个整数,$b\neq 0$,$c\neq 0$,如果 $(a,c)=1$,则

$$(ab,c)=(b,c)$$

定理 1.16　设 p 是素数,若 $p|ab$,则 $p|a$ 或 $p|b$。

定理 1.17　设 $a_1,\cdots,a_n$ 是 n 个整数。如果 $(a_i,c)=1$,$1\leqslant i\leqslant n$,则

$$(a_1\cdots a_n,c)=1$$

定义 1.5(最小公倍数)　设 $a_1,\cdots,a_n$ 是 n 个整数。若 m 是这 n 个数的倍数,则 m 叫做这 n 个数的公倍数。$a_1,\cdots,a_n$ 的所有公倍数中的最小正整数叫做最小公倍数,记作 $[a_1,\cdots,a_n]$。

$m=[a_1,\cdots,a_n]$ 的数学描述是：

(1) $a_1|m,\cdots,a_n|m$。

(2) 若 $a_1|m',\cdots,a_n|m'$,则 $m|m'$。

定理 1.18　设 a,b 是两个互素的正整数,则

(1) 若 $a|m$,$b|m$,则 $ab|m$;

(2) $[a,b]=ab$。

定理 1.19 设 a,b 是两个正整数,则

(1) 若 $a|m$,$b|m$,则$[a,b]|m$;

(2) $[a,b]=ab/(a,b)$。

定理 1.20 设 $a_1,\cdots,a_n$ 是 n 个整数,且 $a_1\neq0$。令

$$[a_1,a_2]=m_2,[m_2,a_3]=m_3,\cdots,[m_{n-1},a_n]=m_n$$

则$[a_1,\cdots,a_n]=m_n$。

定理 1.21(算术基本定理) 任一整数 $n>1$ 都可以表示成素数的乘积,且在不考虑乘积顺序的情况下,该表达式是唯一的,即

$$n=p_1\cdots p_s,\ p_1\leqslant\cdots\leqslant p_s$$

其中 p_i是素数,且若

$$n=q_1\cdots q_t,\ q_1\leqslant\cdots\leqslant q_t$$

其中 q_j是素数,则

$$s=t,\ p_i=q_i,\ 1\leqslant i\leqslant s$$

算术基本定理的"基本"在于将任意整数用素因子乘积进行了表示,从而将对任意整数的性质的研究转化为对其素因子的性质进行研究。这为研究整数的性质提供了一个具有效率的做法。

定理 1.22 任一整数 $n>1$ 可以唯一地表示成

$$n=p_1{}^{\alpha_1}\cdots p_s{}^{\alpha_s},\quad \alpha_i>0,\quad i=1,\cdots,s$$

其中 $pi<pj(i<j)$是素数。该式叫整数 n 的标准分解式。

有了算术基本定理,整除、最大公约数、最小公倍数的求解变得直观了。

定理 1.23 设 n 是一个大于 1 的整数,且有标准分解式

$$n=p_1{}^{\alpha_1}\cdots p_s{}^{\alpha_s},\quad \alpha_i>0,\quad i=1,\cdots,s$$

则 d 是 n 的正因数当且仅当 d 有因数分解式

$$n=p_1{}^{\beta_1}\cdots p_s{}^{\beta_s},\quad \alpha_i\geqslant\beta_i\geqslant0,\quad i=1,\cdots,s$$

定理 1.24 设 a,b 是两个正整数,且都有因数分解式

$$a=p_1{}^{\alpha_1}\cdots p_s{}^{\alpha_s},\quad \alpha_i>0,\quad i=1,\cdots,s$$

$$b=p_1{}^{\beta_1}\cdots p_s{}^{\beta_s},\quad \beta_i\geqslant0,\quad i=1,\cdots,s$$

则 a,b 的最大公因数和最小公倍数分别有因数分解式

$$(a,b)=p_1{}^{\min(\alpha_1,\beta_1)}\cdots p_s{}^{\min(\alpha_s,\beta_s)}$$

$$[a,b]=p_1{}^{\max(\alpha_1,\beta_1)}\cdots p_s{}^{\max(\alpha_s,\beta_s)}$$

推论:设 a,b 是两个正整数,则

$$[a,b](a,b)=ab$$

因为对任意整数 α,β,有

$$\min(\alpha,\beta)+\max(\alpha,\beta)=\alpha+\beta$$

例 1.11 求解(45,100)和[45,100]

解答:易知

$$45=2^0\cdot3^2\cdot5^1$$
$$100=2^2\cdot3^0\cdot5^2$$

由定理 1.24 可知，

$$(45,100)=2^0\cdot3^0\cdot5^1=5$$
$$[45,100]=2^2\cdot3^2\cdot5^2=900$$

思　考　题

[1] 证明：若 $2|n$，$5|n$，$7|n$，那么 $70|n$。

[2] 设 $n\in Z$，证明：$6|(n^3-n)$。

[3] 对每一个奇数 n，证明：$8|(n^2-1)$。

[4] 利用类似于定理 1.4（反证法）和定理 1.5（构造法）的方法证明：$\sqrt{2}$为无理数。

[5] 手动方式求最大公因数：(55,85)，(202,282)。编写程序：Euclid 除法算法计算(a,b)，进行验证。

[6] 手动方式求 s,t，使得 $sa+tb=(a,b)$，其中(a,b)为(202,282)，(1613,3589)，编写程序：扩展的 Euclid 算法求 $sa+tb=(a,b)$，进行算法验证。

[7] 编写程序：利用 Eratosthenes 筛法产生 10000 以内素数。

[8] 形为 $M_p=2^p-1$ 的素数叫做 Mersenne 素数（梅森素数），这里 p 为素数。给出一个计算机程序求前 5 个梅森素数。

[9] 形为 $F_n=2^{2^n}+1$ 的素数为 Fermat 素数（费马素数），给出一个计算机程序证明 F_1,F_2,F_3,F_4 都是素数。

提示：可下载纽约大学的 NTL 算法库，或关注 GIMPS 网址，http://www.mersenne.org/download/freeware.php，截至 2014 年 2 月，已知的梅森素数共有 48 个。从 1997 年至今，所有新的梅森素数都是由互联网梅森素数大搜索（GIMPS）分布式计算项目发现的（http://zh.wikipedia.org/wiki/梅森素数）。

[10] 编写程序对孪生素数的猜想进行实验验证。

[11] 编写程序对哥德巴赫猜想进行实验验证。

[12] 编写程序对 $3x+1$ 猜想进行实验验证。

第 2 章 同　　余

本章介绍整数的另一个重要的二元关系——同余。

本章重点是同余类、简化剩余系、欧拉函数，难点是同余的应用。

2.1 同余和剩余类

同余的概念是通过整除关系给出的。

定义 2.1 给定一个正整数 m，如果两个整数 a，b，有 $m|a-b$，则 a，b 模 m 同余，记作 $a\equiv b \pmod{m}$；否则，a，b 模 m 不同余，记作 $a\not\equiv b \pmod{m}$。

下面的定理给出了同余关系的等价表达形式，即将同余关系表达成一个等式。

定理 2.1 设 m 是一个正整数，a，b 是两个整数，则

$$a\equiv b \pmod{m}$$

的充要条件是存在一个整数 k，使得 $a=b+km$。

下面的定理说明同余是等价关系。

定理 2.2 设 m 是一个正整数，则模 m 同余是等价关系，即

（1）自反性，对任一整数 a，$a\equiv a \pmod{m}$

（2）对称性，若 $a\equiv b \pmod{m}$，则 $b\equiv a \pmod{m}$

（3）传递性，若 $a\equiv b \pmod{m}$，$b\equiv c \pmod{m}$，则 $a\equiv c \pmod{m}$

定理 2.3 设 m 是给一个正整数，$a1$，$a2$，$b1$，$b2$ 是四个整数，如果

$$a1\equiv b1 \pmod{m}$$
$$a2\equiv b2 \pmod{m}$$

则有

$$a1+a2\equiv b1+b2 \pmod{m}$$
$$a1a2\equiv b1b2 \pmod{m}$$

该定理说明同余关系对于加法和乘法是“保持”的。

定理 2.4 设 m 是一个正整数，$ad\equiv bd \pmod{m}$，如果$(d,m)=1$，则

$$a\equiv b \pmod{m}$$

该定理说明了同余关系的“消去”原则是消去值与模互素。

定理 2.5 设 m 是一个正整数，$a\equiv b \pmod{m}$，如果整数 $d|(a,b,m)$，则

$$\frac{a}{d}\equiv\frac{b}{d}\left(\bmod \frac{m}{d}\right)$$

定理 2.6　设 m 是一个正整数，$a \equiv b \pmod{m}$，如果整数 $d|m$，则

$$a \equiv b \pmod{d}$$

上述定理的证明比较简单，留作练习。

例 2.1　已知 2015 年 5 月 4 日是星期一，问从该天算起，第 2^{2015} 天是星期几？

解答：$2^3=8\equiv 1 \pmod{7}$，因此，第 8 天相当于第 1 天。

$2015=3\times 671+2$，$2^{2015}=(2^3)^{671}\times 2^2\equiv 4 \pmod{7}$，即第 2^{2015} 天就相当于第 4 天，为星期五。

因为整数同余关系是一种等价关系，因此全体整数可以按照模 m 是否同余划分成若干两两不相交的集合，使得每一个集合中的任意两个整数模 m 一定同余，而属于不同集合的任意两个整数模 m 不同余。

设 m 是一个正整数，对任意正整数 a，令

$$\mathrm{Ca}=\{c\,|\,c\in Z, c\equiv a \pmod{m}\}$$

Ca 是非空集合，因为 $a\in \mathrm{Ca}$。

定义 2.2　Ca 叫做模 m 的 a 的剩余类。一个剩余类中的任一数叫做该类的剩余或代表元。若 $r_0, r_1, \cdots, r_{m-1}$ 是 m 个整数，且其中任何两个数都不在同一个剩余类里，则 $r_0, r_1, \cdots, r_{m-1}$ 叫做模 m 的一个完全剩余系。完全剩余系 $0,1,2,\cdots,m-1$ 称为最小非负完全剩余系。

模 m 的剩余类通常写成

$$Z/mZ=\{C_0, C_1, \cdots, C_{m-1}\}=\{\mathrm{Ca}\,|\,0\leqslant a\leqslant m-1\}$$

特别的，当 $m=p$ 为素数时，也写成

$$F_p=Z/pZ=\{C_0, C_1, \cdots, C_{p-1}\}=\{\mathrm{Ca}\,|\,0\leqslant a\leqslant p-1\}$$

C_i 也可以记为 $[i]_m$。

例 2.2　取 $m=7$，则模 m 的剩余类为：

$$C_0=[0]_7=\{\cdots,-14,-7,0,7,14,\cdots\}$$
$$C_1=[1]_7=\{\cdots,-13,-6,1,8,15,\cdots\}$$
$$C_2=[2]_7=\{\cdots,-12,-5,2,9,16,\cdots\}$$
$$C_3=[3]_7=\{\cdots,-11,-4,3,10,17,\cdots\}$$
$$C_4=[4]_7=\{\cdots,-10,-3,4,11,18,\cdots\}$$
$$C_5=[5]_7=\{\cdots,-9,-2,5,12,19,\cdots\}$$
$$C_6=[6]_7=\{\cdots,-8,-1,6,13,20,\cdots\}$$

$-14,-6,2,10,18,19,20$ 是模 7 的一组完全剩余系。$0,1,2,3,4,5,6$ 为模 7 的最小非负完全剩余系。

通常情况下，可用 Z_m 表示 m 的最小非负完全剩余系集合，即 $Z_m=\{0,1,2,\cdots,m-1\}$。Z_m 中的加法和乘法都是模 m 意义上的加法和乘法运算。

根据“抽屉原则”，可以得到如下完全剩余系的判定方法：

定理 2.7　设 m 是正整数，则 m 个整数 $a_1, a_2, \cdots, a_m$ 为模 m 的一个完全剩余系的充要条件是它们模 m 两两不同余。

下面给出一个完全剩余系的构造方法：

定理 2.8　设 m 是正整数，整数 a 满足 $(a,m)=1$，b 是任意整数。若 x 遍历模 m 的一个完全剩余系，则 $ax+b$ 也遍历模 m 的一个完全剩余系。

证明：设 $a_1,a_2,\cdots,a_m$ 为模 m 的完全剩余系，由完全剩余系的定义，这组整数模 m 两两不同余。需要证明的是 $aa_1+b,aa_2+b,\cdots,aa_n+b$ 也是模 m 的一组完全剩余系。只需要证明这 m 个数模 m 两两不同余即可(这里用到"抽屉原则")。若存在 a_i 和 a_j，$i\neq j$，使得

$$aa_i+b\equiv aa_j+b \pmod m$$

则有 $m|a(a_i-a_j)$，由于 $(a,m)=1$，所以 $a_i\equiv a_j \pmod m$。这与 $a_1,a_2,\cdots,a_m$ 模 m 两两不同余矛盾。因此，$aa_1+b,aa_2+b,\cdots,aa_n+b$ 模 m 两两不同余。 ■

定理 2.7 中取特例 $b=0$，说明 $aa_1,aa_2,\cdots,aa_n$ 也是模 m 的一组完全剩余系。于是，aa_i 必然有且仅有一个数为 1。于是可引出如下概念。

定义 2.3　设 m 是一个正整数，a 是一个整数，如果存在整数 a' 使得

$$aa'\equiv 1 \pmod m$$

成立，则 a 叫做模 m 的可逆元，a' 叫做 a 的模 m 逆元。

根据定理 2.8，在模 m 的意义下(即 m 的完全剩余系中)，a' 是存在且唯一的。

2.2　简化剩余系，欧拉定理与费马小定理

在模 m 的一个剩余类中，如果有一个数与 m 互素，则该剩余类中所有的数均与 m 互素。

定义 2.4　一个模 m 的剩余类叫做简化剩余类，如果该类中存在一个与 m 互素的剩余。

模 m 的简化剩余类的全体所组成的集合写成：

$$(Z/mZ)^*=\{\mathrm{C}a\,|\,0\leqslant a\leqslant m-1,(a,m)=1\}$$

特别地，当 $m=p$ 且为素数时，也写成

$$F_{p^*}=(Z/pZ)^*=\{C_1,\cdots,C_{p-1}\}=\{\mathrm{C}a\,|\,1\leqslant a\leqslant p-1\}=F_p\backslash\{C_0\}$$

例 2.3　设 $m=12$，则模 m 的简化剩余类为 $\{C_1,C_5,C_7,C_{11}\}$。

例 2.4　设 $m=7$，则模 m 的简化剩余类为 $\{C_1,C_2,C_3,C_4,C_5,C_6\}$。

定义 2.5　设 m 是一个正整数，在模 m 的所有不同简化剩余类中，从每个类任取一个数组成的整数的集合，叫做模 m 的一个简化剩余系(Reduced Residue)。有的书籍也称之为既约剩余系、缩剩余系、缩系。

例 2.5　设 $m=12$，则 1,5,7,11 构成模 12 的简化剩余系。

例 2.6　设 $m=7$，则 1,2,3,4,5,6 构成模 7 的简化剩余系。

定义 2.6　设 m 是一个正整数，则 m 个整数 $0,1,\cdots,m-1$ 中与 m 互素的整数的个数，记为 $\Phi(m)$。叫做欧拉(Euler)函数。

显然，模 m 的简化剩余类的个数为 $\Phi(m)$，即 $|(Z/mZ)^*|=\Phi(m)$。模 m 的简化剩余系的元素个数为 $\Phi(m)$。

与定理 2.7 类似,如下定理给出了简化剩余系的判定方法。

定理 2.9 设 m 是正整数,则 $\Phi(m)$ 个整数 $a_1,a_2,\cdots,a_{\Phi(m)}$ 为模 m 的一个简化剩余系的充要条件是它们与 m 互素,且模 m 两两不同余。

与定理 2.8 类似,有如下定理给出简化剩余系的构造方法:

定理 2.10 设 m 是正整数,整数 a 满足 $(a,m)=1$。若 x 遍历模 m 的一个简化剩余系,则 ax 也遍历模 m 的一个简化剩余系。

证明:因为 $(a,m)=1$,$(x,m)=1$,于是 $(ax,m)=1$。故 ax 为 m 的简化剩余类的剩余。然后只需要证明 $aa_1,aa_2,\cdots,aa_{\Phi(m)}$ 这 $\Phi(m)$ 个数模 m 两两不同余即可。若存在 a_i 和 a_j,$i\neq j$,使得

$$aa_i\equiv aa_j \pmod m$$

则有 $m|a(a_i-a_j)$,由于 $(a,m)=1$,所以 $a_i\equiv a_j \pmod m$。这与 $a_1,a_2,\cdots,a_{\Phi(m)}$ 模 m 两两不同余矛盾。因此,ax 也遍历模 m 的一个简化剩余系。 ■

例 2.7 设 $m=7$,a 为与 m 互素的整数,x 遍历模 m 的简化剩余系,计算 $ax \pmod m$ 可得到下表,从表中可见:ax 也遍历模 m 的一个简化剩余系。

a \ x	1	2	3	4	5	6
1	1	2	3	4	5	6
2	2	4	6	1	3	5
3	3	6	2	5	1	4
4	4	1	5	2	6	3
5	5	3	1	6	4	2
6	6	5	4	3	2	1

定理 2.10 再一次说明了 a 的逆元是存在且唯一的。(第 6 章将会看到,这个表也构成了一个乘法群中的运算表。)

下列定理说明 Euler 函数是乘性函数。

定理 2.11 设 m,n 是互素的两个正整数,则

$$\Phi(mn) = \Phi(m)\Phi(n)$$

例 2.8 $\Phi(77)= \Phi(7)\Phi(11)=6\cdot 10=60$。

例 2.9 $\Phi(30)= \Phi(2)\Phi(3)\Phi(5)=1\cdot 2\cdot 4=8$。

定理 2.12 设正整数 m 的标准分解式为

$$n = \prod_{p|n} p^{\alpha} = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k}$$

则

$$\Phi(n) = n\prod_{p|n}\left(1-\frac{1}{p}\right) = n\left(1-\frac{1}{p_1}\right)\left(1-\frac{1}{p_2}\right)\cdots\left(1-\frac{1}{p_k}\right)$$

证明:当 $m=p^{\alpha}$ 时,模 m 的完全剩余系 $\{0,1,\cdots, p^{\alpha}-1\}$ 的 p^{α} 个整数中,与 p 不互素的只有 p 的倍数,共有 $p^{\alpha-1}$ 个,因此,与 p^{α} 互素的数共有 $p^{\alpha}-p^{\alpha-1}$ 个数,即

$$\Phi(p^{\alpha}) = p^{\alpha} - p^{\alpha-1} = p^{\alpha}(1-1/p)$$

由定理 2.11，

$$\begin{aligned}\Phi(n) &= \Phi(p_1^{\alpha_1})\,\Phi(p_2^{\alpha_2})\cdots\Phi(p_k^{\alpha_k}) \\ &= p^{\alpha_1}(1-1/p_1)\;p^{\alpha_2}(1-1/p_2)\cdots p^{\alpha_k}(1-1/p_k) \\ &= n(1-1/p_1)(1-1/p_2)\cdots(1-1/p_k)\end{aligned}$$

■

例 2.10　计算 $\Phi(120)$。

$120 = 2^3 \cdot 3 \cdot 5$，$\Phi(120) = 120 \cdot (1-1/2) \cdot (1-1/3) \cdot (1-1/5) = 32$

先考查如下例子，看能否归纳出一般规律。

例 2.11　设 $m=12$，$\Phi(12)=4$。1，5，7，11 构成模 12 的简化剩余系，$(5,12)=1$，因此，$5 \cdot 1$，$5 \cdot 5$，$5 \cdot 7$，$5 \cdot 11$ 也构成模 12 的简化剩余系。计算可知

$$5 \cdot 1 \equiv 5 \pmod{12}$$

$$5 \cdot 5 \equiv 1 \pmod{12}$$

$$5 \cdot 7 \equiv 11 \pmod{12}$$

$$5 \cdot 11 \equiv 7 \pmod{12}$$

将上面 4 个式子的左边相乘，由定理 2.3，可得

$$(5 \cdot 1)(5 \cdot 5)(5 \cdot 7)(5 \cdot 11) \equiv 5 \cdot 1 \cdot 11 \cdot 7 \pmod{12}$$

即

$$5^4 \cdot (1 \cdot 5 \cdot 7 \cdot 11) \equiv 5 \cdot 1 \cdot 11 \cdot 7 \pmod{12}$$

由于 $(1 \cdot 5 \cdot 7 \cdot 11, 12) = 1$，因此，由定理 2.4 知，$5^4 \equiv 1 \pmod{12}$，即

$$5^{\Phi(12)} \equiv 1 \pmod{12}$$

思考 2.1　仔细观察上述计算过程，然后尝试给出其他的例子，看能否归纳出一般情况。

定理 2.13　(**Euler 定理**)。设 m 是大于 1 的整数，如果 a 是满足 $(a,m)=1$ 的整数，则

$$a^{\Phi(m)} \equiv 1 \pmod{m}$$

证明：设 $r_1, r_2, \cdots, r_{\Phi(m)}$ 是模 m 的一组简化剩余系，根据定理 2.10

$$ar_1, ar_2, \cdots, ar_{\Phi(m)}$$

也是模 m 的一组简化剩余系，因此

$$(ar_1) \cdot (ar_2) \cdot \cdots \cdot (ar_{\Phi(m)}) \equiv r_1 \cdot r_2 \cdot \cdots \cdot r_{\Phi(m)} \pmod{m}$$

即

$$a^{\Phi(m)} \cdot (r_1 \cdot r_2 \cdots \cdot r_{\Phi(m)}) \equiv r_1 \cdot r_2 \cdot \cdots \cdot r_{\Phi(m)} \pmod{m}$$

又 $(r_1 \cdot r_2 \cdot \cdots \cdot r_{\Phi(m)}, m) = 1$，所以

$$a^{\Phi(m)} \equiv 1 \pmod{m}$$

■

推论 1　(**Fermat 小定理**)。设 p 是一个素数，则对于任意整数 a，均有

$$a^p \equiv a \pmod{m}$$

证明留做练习。

推论 2　若 p 是素数，a 是整数，且 p　a，则 $a \cdot a^{p-2} = 1 \pmod{p}$。(即 a^{p-2} 是 a 模 p 的逆元。)

考查如下例子，看能否归纳出一般规律。

例 2.12 设 $p=17$，是一个素数，观察：

$$
\begin{aligned}
&1\cdot 2\cdot 3\cdot 4\cdot 5\cdot 6\cdot 7\cdot 8\cdot 9\cdot 10\cdot 11\cdot 12\cdot 13\cdot 14\cdot 15\cdot 16\\
&=(2\cdot 9)\cdot(3\cdot 6)\cdot(4\cdot 13)\cdot(5\cdot 7)\cdot(8\cdot 15)\cdot(10\cdot 12)\cdot(11\cdot 14)\cdot(1\cdot 16)\\
&=1\cdot 1\cdot 1\cdot 1\cdot 1\cdot 1\cdot 1\cdot(-1)\\
&=-1 \pmod{17}
\end{aligned}
$$

思考 2.2 仔细观察上述计算过程，然后尝试给出一个类似的例子，看能否归纳出一般情况。

定理 2.14（Wilson 定理） 设 p 是一个素数，则

$$(p-1)!\equiv -1 \pmod{p}$$

证明：若 $p=2$，结论显然成立。

若 $p\geqslant 3$，根据定理 2.10，对于每个整数 a，$1\leqslant a\leqslant p-1$，存在唯一的整数 a'，$1\leqslant a'\leqslant p-1$，使得

$$aa'\equiv 1 \pmod{p}$$

而 $a=a'$ 的充要条件是 a 满足

$$a^2\equiv 1 \pmod{p}$$

此时，$a=1$ 或者 $p-1$。

因此，当 $a\in\{2,3,\cdots,p-2\}$时，有 $a'\in\{2,3,\cdots,p-2\}$，因此，$\{2,3,\cdots,p-2\}$中的 a 和 a' 两两配对。于是，$2\cdot 3\cdot\cdots\cdot(p-2)\equiv 1 \pmod{p}$。

$$(p-1)!\equiv 1\cdot 2\cdot 3\cdot\cdots\cdot(p-2)\cdot(p-1)\equiv 1\cdot(p-1)=-1 \pmod{p} \quad ■$$

图 2.1 给出了证明的示意图。

$$(p-1)!=1\cdot\boxed{2\cdot 3\ldots\cdot(p-2)}\cdot(p-1)=1\cdot(p-1)=-1 \pmod{p}$$

两两配对，乘积为1

图 2.1 Wilson 定理证明的示意图

2.3 模运算和同余的应用

2.3.1 密码系统的基本概念模型

信息安全中最重要的基础部分是密码学，密码学的最初目的是保密。图 2.2 给出了密码系统的基本概念。

定义 2.7 密码系统（Cryptosystem）

一个密码系统是一个五元组 $\langle P,C,K,E,D\rangle$，满足：

（1）P 是可能明文的有限集（明文空间）

（2）C 是可能密文的有限集（密文空间）

（3）K 是一切可能密钥构成的有限集（密钥空间）

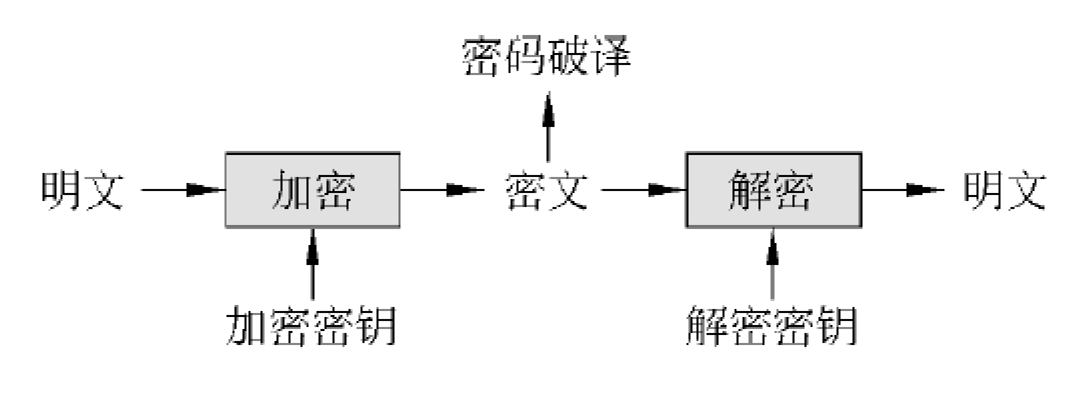


图 2.2　密码系统

(4) 任意 $k\in K$，有一个加密算法 $e_{k_e}\in E$ 和相应的解密算法 $d_{k_d}\in D$，使得 $e_{k_e}:P\to C$ 和 $d_{k_d}:C\to P$ 分别为加密和解密函数，满足 $d_{k_d}(e_{k_e}(x))=x$，这里 $x\in P$。

注解：

古典密码体制通常对字符进行运算，26 个英文字符常被抽象为 0～25 间的整数。(在计算机发明之后，现代密码体制通常对比特 bit 进行运算。)

如果 $k_d=k_e$，即加密密钥和解密密钥，则称为对称密钥密码体制。否则，称为非对称密钥密码体制(又叫公钥密码体制)。古典方案都是对称密钥密码体制，公钥密码密码体制是在 1976 年由 W. Diffie 和 M. Hellman 提出的，它的出现是密码学发展史上的一个里程碑。

对称密码按照明文的类型可分为序列密码(又叫流密码，Stream Cipher)和分组密码(Block Cipher)。序列密码对明文按照字符或者比特逐位加密，对密文逐位解密。分组密码将明文按照一定的长度分组(Block)，加密和解密分组进行。

2.3.2　移位密码

模运算在古典密码中有较多的应用。这是因为古典密码通常对 26 个英文字母进行操作。

将 26 个英文字符从 a～z 依次分别与 0～25 的整数建立一一对应关系。

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

令

$$P=C=K=Z_{26},x\in P,y\in C,k\in K,$$

定义加密解密算法：

$$e_k(x)=x+k \bmod 26$$

$$d_k(y)=y-k \bmod 26$$

例 2.13　Caesar 密码。Caesar 密码是 $k=3$ 的移位密码，若明文为 please，请写出密文。

解答：密文为 sohdvh。

2.3.3　Vigenere 密码

维吉尼亚密码(Vigenere Cipher)于 1858 年由法国密码学家 B. D. Vigenere 提出。

定义 2.8 设 m 为某个固定的正整数，P,C,K 分别为明文空间、密文空间、密钥空间，且 $P=C=K=(Z_{26})^m$，对一个密钥 $k=(k_1,k_2,\cdots,k_m)$，定义：

$$e_k(x_1,x_2,\cdots,x_m)=(x_1+k_1,x_2+k_2,\cdots,x_m+k_m)$$
$$d_k(y_1,y_2,\cdots,y_m)=(y_1-k_1,y_2-k_2,\cdots,y_m-k_m)$$

这里 $(x_1,x_2,\cdots,x_m)$ 为一个明文分组中的 m 个字母。所有运算在 Z_{26} 中进行。密钥长度为 m，故密钥空间为 26^m。明文是按照长度为 m 的分组进行加密的。

Vigenere 密码是典型的多表代换，加密中一个字母可被映射到 m 个可能的字母之一（假定密钥包括 m 个不同的字母），所以分析起来比单表代换更困难。

思考 2.3：设 $m=5$，密钥字为“hello”，如何加密“university”。

明文分组为：unive，rsity，密文转化为 Z_{26} 为：20，13，8，21，4 和 17，18，8，19，24。

密钥字实际为 $k=(7,4,11,11,14)$，Z_{26} 表示的密文为：1，17，19，6，18 和 24，22，19，4，12。密文字母为“brtgs，ywtem”。

2.3.4 Hill 密码

希尔密码(Hill Cipher)由数学家 L. Hill 于 1929 年提出。基本思想是把 m 个连续的明文字母代换成 m 个连续的密文字母，这个代换由密钥决定，这个密钥是一个变换矩阵，解密时只需要对密文做一次逆变换即可。

定义 2.9 设 m 是某个固定的正整数，$P,C,\boldsymbol{K}$ 分别为明文空间、密文空间、密钥空间，且 $P=C=(Z_{26})^m$，设 $\boldsymbol{K}=(k_{ij})_{m\cdot m}$ 是一个 $m\times m$ 可逆矩阵，（即行列式 $\det(\boldsymbol{K})\neq 0$，且 $\gcd(26,\det(\boldsymbol{K})=1)$。对任意密钥 $k\in\boldsymbol{K}$，定义：

$$e_k(x)=x\boldsymbol{K}$$
$$d_k(y)=y\boldsymbol{K}^{-1}$$

所有运算均在 Z_{26} 中进行。特别地，当 $m=1$ 时，Hill 密码退化成单字母仿射密码。

例 2.14 设 $m=2$，密钥 $\boldsymbol{K}=\begin{bmatrix}11 & 8\\ 3 & 7\end{bmatrix}$，容易计算 $\boldsymbol{K}^{-1}=\begin{bmatrix}7 & 18\\ 23 & 11\end{bmatrix}$，设明文为 hill，相应的明文向量为[7,8]和[11,11]，于是，相应的密文向量分别为

$$[7,8]\begin{bmatrix}11 & 8\\ 3 & 7\end{bmatrix}=[77+24,56+56]=[23,8]$$

$$[11,11]\begin{bmatrix}11 & 8\\ 3 & 7\end{bmatrix}=[121+33,88+77]=[24,9]$$

故密文为 xiyj。

思　考　题

[1] 写出模 9 的一个完全剩余系，它的每个数是奇数。

[2] 2015 年 3 月 1 日是星期日，问第 $2^{20150301}$ 天后是星期几？

[3] 写出 $Z/7Z$ 中的加法表和乘法表。

[4] 运用 Wilson 定理，求 8 • 9 • 10 • 11 • 12 • 13 (mod 7)

[5] 编写程序实现 Vigenere 密码算法和 Hill 密码算法。

第 3 章 同余式

第 2 章引入了同余的概念，本章介绍在模 m 情况下同余式的求解。首先介绍一次同余式的求解，再介绍一次同余式组的求解。

本章的重点是中国剩余定理及其在 RSA 中的应用，难点是模重复平方法。

3.1 一次同余式

3.1.1 一次同余式的求解

定义 3.1 设 m 是一个正整数，$f(x)$为多项式且

$$f(x)=a_nx^n+\cdots+a_1x+a_0$$

其中 a_i 是整数，则

$$f(x)\equiv 0 \pmod m$$

称作模 m 同余式。若 $a_n \not\equiv 0 \pmod m$，则 n 叫做 $f(x)$的次数，记为 degf。此时叫做模 m 的 n 次同余式。

如果整数 a 使得

$$f(a)\equiv 0 \pmod m$$

成立，则 a 叫做同余式的解。

事实上，满足 $x\equiv a \pmod m$的所有整数都使得该同余式成立。因此，在讨论同余方程的解时，以一个剩余类作为一个解。在模 m 的完全剩余系中，使得同余式成立的剩余的个数叫做同余式的解数。

下面首先考虑一次同余式的求解。

定理 3.1 一次同余方程 $ax\equiv b \pmod m$有解的充要条件是$(a,m)|b$，且当其有解时，其解数为(a,m)。

证明：先证明必要性。设同余方程 $ax\equiv b \pmod m$有解，解为 x_0，则存在整数 k，使得

$$ax_0\equiv km+b$$

即

$$b = ax_0-km$$

由于$(a,m)|a$，$(a,m)|m$，因此

$$(a,m) \mid ax_0-km=b$$

再证明充分性。

设 $a'=\frac{a}{(a,m)}$，$m'=\frac{m}{(a,m)}$，$b'=\frac{b}{(a,m)}$，易知，a'，m'，b' 均为整数。

首先考虑同余方程

$$a'x\equiv 1 \pmod{m'}$$

因为 $\gcd(a',m')=1$，可知 a' 存在模 m' 的乘法逆元 x_0（见定义 2.3）。满足 $a'x_0\equiv 1 \pmod{m'}$，且在模 m' 下，逆元是唯一的，即同余方程 $a'x\equiv 1 \pmod{m'}$ 存在唯一解

$$x\equiv x_0 \pmod{m'}$$

因此，易知同余方程

$$a'x\equiv b' \pmod{m'}$$

也存在唯一解

$$x\equiv x_1\equiv x_0 b' \pmod{m'}$$

下面证明这个解也是 $ax\equiv b \pmod{m}$ 的特解：

不妨设 $x=k_1 m'+x_0 b'$，$k_1\in z$

$$ax=ak_1 m'+ax_0 b'=ak_1\frac{m}{(a,m)}+ax_0\frac{b}{(a,m)}=a'k_1 m+a'x_0 b$$
$$\equiv a'x_0 b \pmod{m}$$

由于 $a'x_0\equiv 1 \pmod{m'}$，不妨设 $a'x_0=k_2 m'+1$，$k_2\in z$

$$a'x_0 b=(k_2 m'+1)b=k_2 m'b+b=k_2\frac{m}{(a,m)}b+b=k_2 m b'+b$$
$$\equiv b \pmod{m}$$

最后，$ax\equiv b \pmod{m}$ 的全部解为：

$$x\equiv x_1+t\frac{m}{(a,m)} \pmod{m},\quad t=0,1,\cdots,(a,m)-1$$

其原因是：

如果同时有同余式

$$ax\equiv b \pmod{m} \text{ 和 } ax_1\equiv b \pmod{m}$$

成立，两式相减得到

$$a(x-x_1)\equiv 0 \pmod{m}$$

这等价于

$$x\equiv x_1\left(\bmod \frac{m}{(a,m)}\right)$$

因此，x 只要与 x_1 关于 $\frac{m}{(a,m)}$ 同余，即为 $ax\equiv b \pmod{m}$ 的解。 ■

定理 3.1 的证明过程其实给出了一次同余式求解的过程。

定理 3.2 设 m 是一个正整数，a 是满足 $(a,m)|b$ 的整数，则一次同余式

$$ax\equiv b \pmod{m}$$

的全部解为

$$x \equiv \left(\left(\frac{a}{(a,m)}\right)^{-1}\left(\bmod \frac{m}{(a,m)}\right)\right)\frac{b}{(a,m)} + t\frac{m}{(a,m)} \pmod m$$

$$t = 0,1,\cdots,(a,m)-1$$

定理 3.2 其实给出了一次同余式“根与系数的关系”。定理 3.2 是定理 3.1 的充分性证明过程的结果，图 3.1 给出了解的 3 个部分与定理 3.1 的对应关系。为便于记忆，这里将这 3 个部分简称为：“模系收缩求逆元”、“逆元扩大求特解”、“t 变模扩求全解”。“模系收缩”的目的是使得“系数 a”和“模 m”之间互素，从而可以求“系数 a 的逆元”。有了“逆元”之后就可以扩大“$b'=\frac{b}{(a,m)}$”求出特解。最后再由特解求全解。

$$x \equiv \underbrace{\underbrace{\underbrace{\left(\left(\frac{a}{(a,m)}\right)^{-1}\left(\bmod \frac{m}{(a,m)}\right)\right)}_{x_0}\frac{b}{(a,m)}}_{x_0 b'\ (x_1)} + t\frac{m}{(a,m)}}_{x_1+t\frac{m}{(a,m)}} \pmod m \quad t=0,1,\cdots,(a,m)-1$$

图 3.1　一次同余式“根与系数的关系”示意图

由定理 3.2 可以给出一个求解一次同余式的算法。

例 3.1　求解同余方程 $32x \equiv 12 \pmod 8$。

解答：$(32,8) \nmid 12$，由定理 3.1 知，同余方程无解。

例 3.2　求解同余方程 $6x \equiv 2 \pmod 8$。

解答：先判断 $(6,8) \mid 2$，所以同余方程有解。

先解同余方程 $3x \equiv 1 \pmod 4$，此方程解唯一，易知其解为

$$x \equiv 3 \pmod 4$$

取 $x_0 = 3$，则

$$x = x_0 + \frac{8}{2}t = 3 + 4t, \quad (t = 0,1)$$

为原方程的解。原方程的所有解为

$$x \equiv 3 \pmod 8$$

$$x \equiv 3+4 \equiv -1 \pmod 8$$

3.1.2　一次同余式在仿射加密中的应用

仿射密码是一种古典密码，其算法设计时用到的数学基础是模运算和同余方程。上一章曾经介绍过移位密码，由于移位密码的密钥量太小，且移位密码在加密代换后字母的先后次序其实没有改变。仿射密码可以改进上述两个弱点。

定义明文空间 $P=Z_{26}$、密文空间 $C=Z_{26}$、密钥空间为：

$$K=\{(a,b)\in Z_{26}\cdot Z_{26}:\gcd(a,26)=1\}$$

对于

$$x\in P, y\in C, k=(a,b)\in K$$

定义加密函数：

$$e_k(x)=ax+b \bmod 26$$

解密函数：

$$d_k(y)=a^{-1}(y-b) \bmod 26$$

例 3.3 利用仿射加密，$a=3$，$b=5$，模为 26。

解答：易知，加密函数为 $e(x)=3x+5 \pmod{26}$

加密明文“hello”，其在 Z_{26} 表示为 7，4，11，11，14。

加密密文用 Z_{26} 表示为 0，17，12，12，21。即密文为“armmv”。

思考 3.1 仿射密码中 a 是否有要求。

根据定理 3.1，这里要求 $(a,26)=1$，否则，加密函数就不是一个单射函数。

例如：当 $k=(6,1)$ 时，$(a,26)=(6,26)=2$ 时，对 $x\in Z_{26}$，有

$$6(x+13)+1=6x+1 \bmod 26$$

于是 x，$x+13$ 都是 $6x+1$ 的明文。

思考 3.2 证明 $(a,26)=1$ 时，仿射密码的解唯一。

证明：设存在 $x_1,x_2\in Z_{26}$，使得 $e_k(x)=ax_1+b=ax_2+b \bmod 26$，于是 $ax_1=ax_2 \bmod 26$，有 $26|a(x_1-x_2)$，又因为 $(a,26)=1$，所以 $26|(x_1-x_2)$，由于 $x_1,x_2\in Z_{26}$，得到 $x_1=x_2$。 ■

思考 3.3 仿射密码的密钥空间大小，即密钥的数量有多少。

a 的可能性为 12，因为 $a\in Z_{26}$，$\gcd(a,26)=1$，即

$$a=\phi(26)=\phi(2\cdot 13)=\phi(2)\cdot\phi(13)=1\cdot 12=12$$

$b\in Z_{26}$，b 的可能性为 26。

故整个密钥空间大小为 $12\cdot 26=312$。

如果密钥空间太小，容易导致穷举所有可能的密钥，然后看能否解密密文的攻击。

另外，当 $a=1$ 时，仿射密码就退化为移位密码。

3.2 中国剩余定理

在研究了一次同余式之后，下面考虑一次同余式组。

中国剩余定理(Chinese Remainder Theorem，CRT)，又称孙子定理，最早见于公元5～6世纪，我国南北朝的一部经典数学著作《孙子算经》中的“物不知数”问题：

“今有物不知其数，三三数之剩二，五五数之剩三，七七数之剩二，问物几何？”

这其实是求解一个一次同余方程式组，即

$$\begin{cases} x\equiv 2 \bmod 3 \\ x\equiv 3 \bmod 5 \\ x\equiv 2 \bmod 7 \end{cases}$$

《孙子算经》中给出了解法，但这只是一个孤立的例子。南宋数学家秦九韶创立的“大衍求一术”一般性地解决了一次同余方程组的求解问题，中国古代这一成果统称为“孙子定理”，在外国文献中常被称为“中国剩余定理”。它可能是最著名的由中国人给出的

算法。

在给出算法之前，请先体会一下如下同余式组：

$$\begin{cases} x \equiv 2 \bmod 3 \\ x \equiv 2 \bmod 5 \\ x \equiv 2 \bmod 7 \end{cases}$$

容易知道，这个问题容易解决，即 $x \equiv 2 \pmod{105}$，于是 $x=3\cdot5\cdot7K+2, K\in Z$。

问题是，如果余数不再相等，则问题就有点麻烦了，再来看同余式组：

$$\begin{cases} x \equiv 2 \bmod 3 \\ x \equiv 3 \bmod 5 \\ x \equiv 4 \bmod 7 \end{cases}$$

设法从 $3K_1+2, 5K_2+3, 7K_3+4 (K_1, K_2, K_3 \in Z)$ 中找到同一个数看上去是不容易的。也就是说，找到一个数同时满足这 3 个条件是不容易的。于是，能否换个思路，即能否把 x“想象”成由 3 个部分组成。为便于对 CRT 求解过程的理解，下面专门给出一个特别的表述和解释。

写成 3 个部分的好处是：每个部分的条件将“更容易”满足。

令 $x=A+B+C$，若 A, B, C 满足如下条件，则为解。这里 $[a]_m$ 表示模 m 与 a 同余。

$$A=[2]_3, B=[0]_3, C=[0]_3$$
$$A=[0]_5, B=[3]_5, C=[0]_5$$
$$A=[0]_7, B=[0]_7, C=[2]_7$$

看最左边一列，A 为 $5\cdot7$ 的倍数，且模 3 余 2，这样 A 就比较容易求解了。A 可用如下方法计算：

$A=(5\cdot7)\cdot(5\cdot7)^{-1}\cdot2$。因为，这个表达式中有 $(5\cdot7)$，因此是 5 和 7 的倍数。同时这个表达式模 3 余 2。原因是：$(5\cdot7)=35$。$(5\cdot7)^{-1}$ 表示 $(5\cdot7)$ 模 3 的逆元。35 模 3 的逆元是 2。两者相乘必然模 3 余 1。因此再乘以 2 后必然模 3 余 2。即 $A=35\cdot2\cdot2=140$ 模 3 余 2，且为 5 和 7 的倍数。

同理，B 为 $3\cdot7$ 的倍数，且模 5 余 3，于是，可用如下方法计算：$B=(3\cdot7)\cdot(3\cdot7)^{-1}\cdot3$，这里 $(3\cdot7)^{-1}$ 表示 $(3\cdot7)$ 模 5 的逆元。$3\cdot7=21$，21 模 5 的逆元是 1。因此，$B=21\cdot1\cdot3=63$。

同理，C 为 $3\cdot5$ 的倍数，且模 7 余 2，于是，可用如下方法计算：$C=(3\cdot5)\cdot(3\cdot5)^{-1}\cdot2$，这里 $(3\cdot5)^{-1}$ 表示 $(3\cdot5)$ 模 7 的逆元。$3\cdot5=15$，15 模 7 的逆元是 1。因此，$C=15\cdot1\cdot2=30$。

于是 $x=A+B+C=140+63+30=233$。又 $233 \bmod 105 = 23$。因此，所有解为 $105K+23$。

在程大位著的《算法统要》(1593 年)，用四句诗给出了上述解答过程的中几个关键数字：

三人同行古来稀，五数梅花廿一枝；
七子团圆正月半，除百零五便得之。

这几个关键的数字是：与模 3 对应的是 70(“古来稀”)，即 $35\cdot2$；与模 5 对应的是 21

(“廿一枝”),即 21·1;与模 7 对应的是 15·1=15(“正月半”)。有了这三个关键数字,只要给出 x 模 3,5,7 的余数 a_1,a_2,a_3,即可以计算出结果 $x=a_1\cdot 70+a_2\cdot 21+a_3\cdot 15 \pmod{105}$。结果除去 105=3·5·7 的倍数(“除百零五”)即可以得到答案。

一般地,如果 m_1,m_2,m_3 是两两互素的正整数,对于同余方程组

$$\begin{cases}x\equiv a_1 \bmod m_1\\ x\equiv a_2 \bmod m_2\\ x\equiv a_3 \bmod m_3\end{cases}$$

令 $m=\prod_{i=1}^{3} m_i$, $M_i=m/m_i$,则解为

$$x\equiv\prod_{i=1}^{3} M_i\cdot M_i^{-1}\cdot a_i \pmod m$$

当然,上述解法可以推广到一般情况。

例 3.4 韩信点兵问题:有兵一队,若列成五行纵队,则末行一人,若列成六行纵队,则末行五人;若列成七行纵队,则末行四人;若列成十一行纵队,则末行十人。求兵数。

解答:转化为同余式组

$$\begin{cases}x\equiv 1 \bmod 5\\ x\equiv 5 \bmod 6\\ x\equiv 4 \bmod 7\\ x\equiv 10 \bmod 11\end{cases}$$

应用 CRT,$m=5\cdot 6\cdot 7\cdot 11=2310$

$$M_1=2310/5=462, M_1^{-1}=3 \pmod 5$$
$$M_2=2310/5=385,\ M_2^{-1}=1 \pmod 6$$
$$M_3=2310/5=330,\ M_3^{-1}=1 \pmod 7$$
$$M_4=2310/11=210,\ M_4^{-1}=1 \pmod{11}$$
$$x\equiv 462\cdot 3\cdot 1+385\cdot 1\cdot 5+330\cdot 1\cdot 4+210\cdot 1\cdot 10$$
$$\equiv 6731\equiv 2111 \pmod{2310}$$

下面给出 CRT 的严格证明。

定理 3.3(中国剩余定理) 设 $m_1,m_2,\cdots,m_k$ 是 k 个两两互素的正整数,令 $m=\prod_{i=1}^{k} m_i$, $M_i=m/m_i$,$(i=1,2,\cdots,k)$,则对任意的整数 $a_1,a_2,\cdots,a_k$,同余式组

$$\begin{cases}x\equiv a_1 \bmod m_1\\ x\equiv a_2 \bmod m_2\\ \cdots\\ x\equiv a_k \bmod m_k\end{cases}\tag{3.1}$$

有唯一解

$$x\equiv\prod_{i=1}^{k} M_i\cdot M_i^{-1}\cdot a_i \pmod m\tag{3.2}$$

其中 $M_iM_i^{-1}\equiv 1 \pmod{m_i}$

证明：由于$(m_i,m_j)=1, i\neq j$，即得$(M_i, m_i)=1$，由定义 2.3 和定理 2.8 知，对每一个 M_i，有一个 M_i^{-1} 存在，使得 $M_iM_i^{-1}\equiv 1 \pmod{m_i}$。另外，由于 $m=m_iM_i$，因此，$m_j|M_i, i\neq j$，故

$$\prod_{i=1}^{k} M_i \cdot M_i^{-1} \cdot a_i \equiv M_i \cdot M_i^{-1} \cdot a_i \equiv a_i \pmod{m_i} \quad i=1,2,\cdots,k$$

即(3.2)为(3.1)的解。

若 x_1, x_2 是满足(3.2)的任意两个整数，则 $x_1\equiv x_2 \pmod{m_i}$$(i=1,2,\cdots,k)$，因为$(m_i,m_j)=1, i\neq j$，于是 $x_1\equiv x_2 \pmod{m}$，故式(3.1)仅有解式(3.2)。■

根据 CRT，可以给出一个求解一次同余式组的算法。

3.3 同余式的应用

3.3.1 RSA 公钥密码系统

一个公钥密码系统的示意如图 3.2 所示，具体的描述如下。

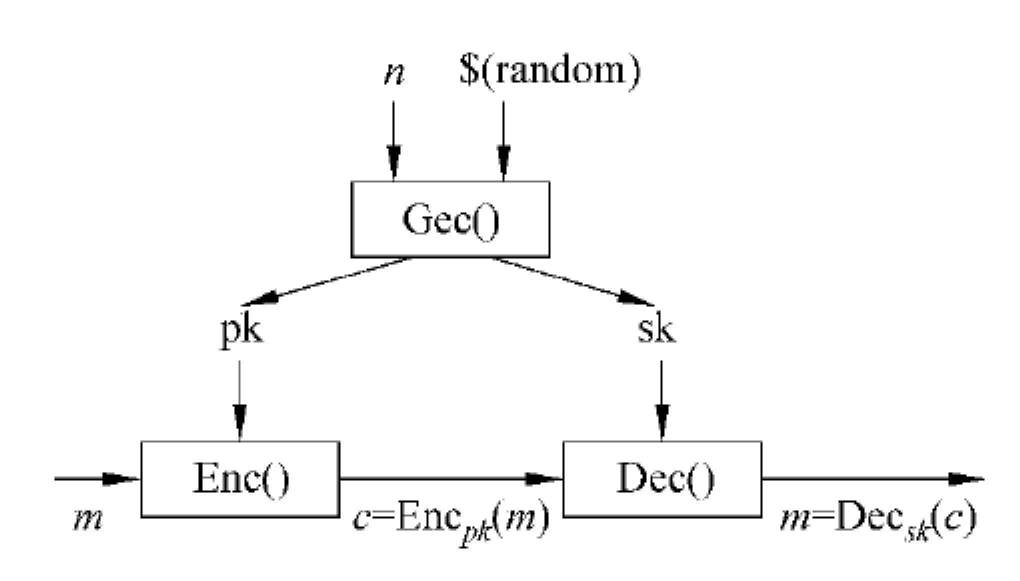


图 3.2　公钥加密体制的示意图

定义 3.2　一个公钥加密体制是这样的一个 6 元组$(M, C, K, \text{Gen}(), \text{Enc}_{pk}(), \text{Dec}_{sk}())$，满足如下条件：

(1) M 是可能消息的集合。

(2) C 是可能密文的集合。

(3) 密钥空间 K 是一个可能密钥的有限集。

(4) 密钥生成算法 Gen()：输入安全参数，输出公钥 pk 和私钥 sk。

(5) 加密算法 $\text{Enc}_{pk}()$：根据输入的公钥 pk 和明文 m，输出密文 $c=\text{Enc}_{pk}(m)$。

(6) 解密算法 $\text{Dec}_{sk}()$：根据输入的私钥 sk 和密文 c，输出明文 $m=\text{Dec}_{sk}(c)$。

注解：

(1) 与对称加密体制的一个最大的不同是加密密钥和解密密钥是不同的，且加密密钥可以公开，解密密钥需要保密。

(2) 密钥生成算法在对称加密体制中是没有的(或者说是平凡的)，而在公钥密码体制中却是不平凡的。密钥生成算法可能是随机生成的密钥。

(3) 随机性。加密算法可能是随机的，即在算法中可以使用一个随机数，输出的密文

与该随机数有关，这种算法叫做概率加密。（例如 5.4 节的 ElGamal 加密。）

（4）确定性。解密算法一定是确定的。

（5）安全性（单向性）：在已知密文 c 和公钥 pk 的情况下，推出明文 m 在计算上是不可行的。对于任意的 $k \in K$，在已知 $\mathrm{Enc}_{pk}()$ 的情况下推出 $\mathrm{Dec}_{sk}()$ 是计算不可行的。对于任意的 $k \in K$，在已知 pk 的情况下，推出 sk 是计算不可行的。

（6）有效性（实用性）：公钥和私钥的产生，即密钥生成是容易的。即 Gen() 是多项式时间内可计算的。已知公钥 pk 和明文 m，计算密文 $c=\mathrm{Enc}_{pk}(m)$ 也是多项式时间可计算的。

（7）一致性。对每一个 $k=(pk,sk)\in K$，都对应一个加密算法 $\mathrm{Enc}_{pk}: M \rightarrow C$ 和解密算法 $\mathrm{Dec}_{sk}(): C \rightarrow M$，满足对于任意的 $m \in M$，若 $c=\mathrm{Enc}_{pk}(m)$，则 $\mathrm{Dec}_{sk}(c)=m$。

（8）陷门性。若已知私钥 sk，存在多项式时间算法可以有密文 c 计算出明文 m，满足 $c=\mathrm{Enc}_{pk}(m)$，其中私钥 sk 称为陷门信息（trapdoor information）。

1977 年 MIT 的 Ronald L Rivest、Adi Shamir、Leonard Adleman 在 MIT 的技术报告中提出 RSA 方案，正式发表于 1978 年的 *Communication of ACM* 期刊上。①

RSA 利用了单向陷门函数的原理，其示意图如图 3.3 所示，陷门信息是解密密钥即私钥 d（与之类似的陷门是 n 的分解，因为如果知道 n 的分解 p 和 q，就可以从公钥求出私钥 d）。

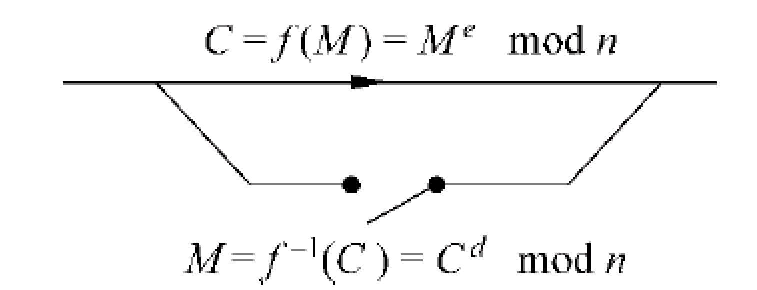


图 3.3　RSA 利用单向陷门函数的原理示意图

RSA 公钥密码方案描述如下：

1. 密钥生成

（1）选取两个大素数 p 和 q（例如长度都接近 512bit）；

（2）计算乘积 $n=p \cdot q$，$\Phi(n)=(p-1)(q-1)$，其中 $\Phi(n)$ 为 n 的欧拉函数；

（3）随机选择整数 $e(1<e<\phi(n))$，要求满足 $\gcd(e,\phi(n))=1$，即 e 与 $\phi(n)$ 互素。

（4）用扩展的 Euclidean 算法计算私钥 d，以满足 $d \cdot e \equiv 1 \mod (\phi(n))$，即 $d \equiv e^{-1} \mod (\phi(n))$。

公钥为 e 和 n，d 是私钥。（两个素数 p 和 q，可销毁，不能泄露。）

2. 加密过程

明文先转换为比特串分组，使每个分组对应的十进制数小于 n，即分组长度小于

① RSA 是第一个实用的公钥密码系统，是目前应用最广泛的公钥密码系统。后来，三位发明者在 2002 年获得了计算机领域的最高奖项——ACM 图灵奖。文献见 R. Rivest, A. Shamir, L. Adleman (1978). A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. Communications of the ACM 21 (2): 120-126。

$\log_2 n$，然后对每个明文分组 m_i 作加密运算，具体过程如下：

(1) 获得接收公钥 (e,n)。

(2) 把消息 M 分组长度为 $L(L<\log_2 n)$ 的消息分组 $M=m_1m_2\cdots m_t$。

(3) 使用加密算法 $c_i=m_i^e \bmod n(1\leqslant i\leqslant t)$，计算出密文 $c=c_1c_2\cdots c_t$。

3. 解密过程

(1) 将密文 c 按长度 L 分组得 $c=c_1c_2\cdots c_t$。

(2) 使用私钥 d 和解密算法 $m_i=c_i^d \bmod n(1\leqslant I\leqslant t)$ 计算 m_i。

(3) 得明文消息 $M=m_1m_2\cdots m_t$。

解密算法的正确性证明如下：

$$c_i^d \bmod n\equiv m_i^{ed} \bmod n\equiv m_i^{k\phi(n)+1} \bmod n$$

分两种情况讨论：

(1) $\gcd(m_i,n)=1$。由欧拉定理，得

$$m_i^{\phi(n)}\equiv 1 \bmod n,\ m_i^{k\phi(n)}\equiv 1 \bmod n,\ m_i^{k\phi(n)+1}\equiv m_i \bmod n$$

于是 $c_i^d \bmod n\equiv m_i \bmod n$。

(2) $\gcd(m_i,n)\neq 1$。由于 $n=p\cdot q$，$\gcd(m_i,n)|n$，所以 $\gcd(m_i,n)=p$ 或 q。

不妨设 $\gcd(n,m_i)=p$，$p|m_i$，令 $m_i=sp$，$1\leqslant s\leqslant q$。

① $\gcd(m_i,q)=1$，由 Fermat 定理可得 $m_i^{q-1}\equiv 1 \bmod q$，于是 $(m_i^{q-1})^{k(p-1)}\equiv 1 \bmod q$，即 $m_i^{k\phi(n)}\equiv 1 \bmod q$。

② 另外，由 $p|m_i$，得 $m_i^{ed}\equiv 0\equiv m_i \bmod p$，故 $m_i^{ed}\equiv m_i \bmod p$。

由①②，且 $\gcd(p,q)=1$。由中国剩余定理，$m_i^{ed}\equiv m_i \bmod n$，于是，$c_i^d \bmod n\equiv m_i \bmod n$。

综合(1)和(2)，有 $c_i^d \bmod n\equiv m_i \bmod n$。又 $m_i<n$，故 $m_i \bmod n=m_i$。 ■

例 3.5 取 $p=11$，$q=13$，那么 $n=pq=11\cdot 13=143$，$\phi(n)=(p-1)(q-1)=120$，选取 $e=17$，满足 $\gcd(e,\phi(n))=\gcd(17,120)=1$。

使用扩展的 Euclidean 算法计算 $d=e^{-1}=113 \bmod 120$，所以公钥为 $(n,e)=(143,17)$，私钥为 $d=113$。

假设对明文 $m=24$ 进行加密，密文为 $c\equiv m^e\equiv 24^{17}\equiv 7 \bmod 143$。密文 $c=7$ 经公开信道发送到接收方后，接收方用私钥 $d=113$ 对密文解密：$m\equiv c^d\equiv 7^{113}\equiv 24 \bmod 143$。从而恢复明文。

3.3.2 CRT 在 RSA 中的应用

(1) 利用 CRT 进行运算加速。

解密者在计算 $m=c^d(\bmod n)$ 的过程中，可以分别计算 $m=c^d(\bmod p)$ 和 $m=c^d(\bmod q)$。然后利用 CRT 计算出 m。

例 3.6 计算 $312^{13}(\bmod 667)$

解答： 令 $x=312^{13}$，解密者知道 $667=23\cdot 29$，所以计算等价于求解同余式组

$$\begin{cases} x\equiv a_1 \bmod 23 \\ x\equiv a_2 \bmod 29 \end{cases}$$

利用模重复平方法(将在 3.3.3 节介绍),得到 $a_1=312^{13}\pmod{23}=8$,$a_2=312^{13}\pmod{29}=4$。再利用 CRT,有

$$M_1=29, M_1^{-1}=4\pmod{23}, M_2=23, M_2^{-1}=-5\pmod{29}$$

$$x\equiv 4\cdot 29\cdot 8+(-5)\cdot 23\cdot 4=468\pmod{667}$$

思考 3.4:为什么 CRT 只能对解密加速,不能对加密加速。

解密需要知道 n 的分解,即需要知道 p,q,有 $n=pq$。只有解密者知道 p,q。

另外,因为解密可以加速,并且加密的速度对用户体验的影响可能更大,所以很多情形下 RSA 加密密钥 e 较解密密钥 d 小。

(2) 利用 CRT 进行低加密指数攻击。

如果加密密钥中加密指数 e 很小,例如 $e=3$,设明文为 m,密文 $c\equiv m^3 \bmod n$,如果 m 较小,则 c 有可能小于 n,则 mod n 操作未起作用,故可对 c 直接开 3 次方得到 m。如果 m 较大,mod n 操作起了作用,虽然不能直接开方,仍然有可能得到 m。假设有 3 个用户接收 m,模数分别为 n_1,n_2,n_3。设明文为 m,密文分别是:

$$c_1\equiv m^3 \bmod n_1$$

$$c_2\equiv m^3 \bmod n_2$$

$$c_3\equiv m^3 \bmod n_3$$

一般 $\gcd(n_i,n_j)\neq 1, i\neq j, i,j\in\{1,2,3\}$,否则可通过 $\gcd(n_i,n_j)$ 得到 n_i,n_j 的分解(从而泄露了 p,q),于是由中国剩余定理,可从三个密文同余式求出 $m^3 \bmod n_1n_2n_3$。由于此时 $0<m^3\leqslant n_1n_2n_3$,可直接对 m^3 开立方得到 m。

推而广之,若加密指数为 e,则得到相同明文的 e 个密文即可由该攻击方法恢复出明文。因此,同一消息加密后发送给多个实体时(若此时的加密指数相同),不要使用小的加密指数。

3.3.3 模重复平方算法

RSA 加密和解密时都需要计算模幂。平凡的求模幂方法在幂指数较大时耗时非常长。因此,寻求快速求模幂的算法对于 RSA 的加密解密效率至关重要。模重复平方(也称为平方乘)算法可以加快计算模幂的速度。下面介绍该算法。

要计算 $c=m^e \bmod n$,不妨设 e 的二进制表示为

$$\begin{aligned} e &= e_{k-1}2^{k-1}+e_{k-2}2^{k-2}+\cdots+e_1 2^1+e_0 \\ &= 2(2(\cdots(2(2(e_{k-1})+e_{k-2})+)\cdots)+e_1)+e_0 \end{aligned}$$

于是

$$\begin{aligned} c &\equiv m^e \bmod n \\ &\equiv m^{e_{k-1}2^{k-1}+e_{k-2}2^{k-2}+\cdots+e_1 2^1+e_0} \bmod n \\ &\equiv ((\cdots\underbrace{\underbrace{((m^{e_{k-1}})^2}m^{e_{k-2}})^2}\cdots m^{e_2})^2 m^{e_1})^2 m^{e_0} \bmod n \end{aligned}$$

从表达式可以看到,如果 $e_i=1$,则 m 即模将需要平方,并且反复进行,因而,称为“模重复平方法”。

根据这一表达式,可以设计计算模幂的快速算法。

算法 3.1：Square-and-Multiply(m,e,n)。

```
/* 模重复平方(平方乘)算法,计算 c=m^e mod n。 */
/* 输入:m,幂次 e,模 n                      */
/* 输出:模幂的结果 c                        */
{
c=1
for i=k-1 to 0
{
    c = c^2 mod n
    If (e_i==1)   c ← c·m mod n
}
Return c
}
```

可以看到，在算法中 $i=k-1$ 时，即首次进入循环时，总满足条件 $e_{k-1}=1$，$c=1 \cdot m \bmod n$。然后，每次进入循环后，先平方，从 e 的高位向低位考查，若该位为 1，则乘上 m，否则不乘。最后一次进入循环，$i=0$ 时，若 $e_0=1$，则乘上 m，否则不乘。

例 3.7　计算 $9726^{3533} \bmod 11413$。

解答：$3533=(110111001101)_2$，$m=9726$，表 3.1 给出了计算过程。

表 3.1　模重复平方法的计算过程

i	e_i	c
11	1	$1^2 \cdot 9726=9726$
10	1	$9726^2 \cdot 9726=2659$
9	0	$2659^2=5634$
8	1	$5634^2 \cdot 9726=9167$
7	1	$9167^2 \cdot 9726=4958$
6	1	$4958^2 \cdot 9726=7783$
5	0	$7783^2=6298$
4	0	$6298^2=4629$
3	1	$4629^2 \cdot 9726=10185$
2	1	$10185^2 \cdot 9726=105$
1	0	$105^2=11025$
0	1	$11025^2 \cdot 9726=5761$

思考 3.5：计算过程中有几次平方，几次乘法。

模平方运算有 12 次，模乘法运算有 8 次。平均而言，有 $\log_2 e$ 次模平方运算和约 $0.5\log_2$ 次，模平方和模乘法均视为模乘法运算，则总计算次数约为 $1.5\log_2 e$ 次模乘法，最

多不超过 $2\log_2 e$ 次模乘法运算。

算法的效率决定了 RSA 加密和解密是否实用。下面给出一个较严格的算法分析。首先来看几个基本模运算的效率，即在 Zn 中的运算，假定 n 为一个 l 比特的整数($l=\log_2 n$)，$0\leqslant m_1, m_2\leqslant n-1$。设 c 为一个正整数，结果如表 3.2 所示。

表 3.2 基本模运算的时间复杂度

运　　算	时间复杂度	运　　算	时间复杂度
$(m_1+m_2) \bmod n$	$O(l)$	$(m_1 m_2) \bmod n$	$O(l^2)$
$(m_1-m_2) \bmod n$	$O(l)$	$(m_1)^{-1} \bmod n$	$O(l^2)$

下面分析模重复平方算法(算法 3.2)的效率，有 k 次循环(即指数 e 的二进制位数)，每次循环中总要执行 1 次平方运算(视为模乘)，或者加上 1 次的模乘，故 1 次循环执行 2 次模乘或 1 次模乘。k 次循环最多 $2k$ 次模乘。根据表中所示模乘的时间复杂度为 $O(l^2)$，l 为模 n 的长度(即 $l=\log_2 n$)，故总时间复杂度为 $O(kl^2)$。另外，通常 $e<n$，故 $k<1$，于是时间复杂度为 $O(l^3)=O((\log_2 n)^3)$。因此，RSA 的加密(或解密)都是在关于明文(或密文)的比特长度的多项式时间内完成。

另外，由于模重复平方算法的循环中模乘的次数等于加密密钥 e 的二进制表示中“1”的个数，故选择二进制表示中“1”较少的那种加密密钥 e 将会加快 RSA 加密的速度，例如 $e=2^{16}+1$，循环中只有 2 次模乘。

思　考　题

[1] 计算 $2^{1000000}\pmod{1309}$(提示：CRT+欧拉定理)。

[2] 编写程序实现中国剩余定理(CRT)，以韩信点兵为例(韩信带 1500 名兵士打仗，战死四五百人，站 3 人一排，多出 2 人；站 5 人一排，多出 4 人；站 7 人一排，多出 6 人。韩信马上说出人数：1049)。

[3] 程序设计：模重复平方法计算 $x^n \pmod{m}$ 。计算 RSA，明文 $m=53$，公钥 $e=35$，$n=31\cdot 37$。

[4] 计算 $2^{32}\pmod{47}$，$2^{43}\pmod{71}$，并用程序来验算。

[5] 编写程序实现 300 十进制位 RSA 公钥密码系统。

第 4 章 二次同余式和平方剩余

第 3 章讨论了一次同余式，本章讨论二次同余式。

本章重点是平方剩余的判定，Legendre 符号及其计算方法。难点是 Rabin 公钥密码系统。

4.1 二次同余式和平方剩余

二次同余式的一般形式是：

$$ax^2+bx+c\equiv 0 \pmod m$$

其中 $a\not\equiv 0 \pmod m$

因为正整数 m 有素因子分解式 $m=p_1^{\alpha_1}\ p_2^{\alpha_2}\cdots p_k^{\alpha_k}$，所以二次同余式等价于同余式组

$$\begin{cases} ax^2+bx+c\equiv 0 \pmod{p_1^{\alpha_1}} \\ \cdots \\ ax^2+bx+c\equiv 0 \pmod{p_k^{\alpha_k}} \end{cases}$$

因此，只需要讨论模为素数幂 p^α 的同余式

$$ax^2+bx+c\equiv 0 \pmod{p^\alpha},\quad p\quad a$$

通过配方，可以进一步变形为

$$(2ax+b)^2\equiv b^2-4ac \pmod{p^\alpha}$$

令 $y=2ax+b$，有

$$y^2\equiv b^2-4ac \pmod{p^\alpha}$$

因此，重点关注如下形式的二次同余式：

$$x^2\equiv a \pmod{p^\alpha}$$

定义 4.1：设 m 为正整数，若同余式

$$x^2\equiv a \pmod m,\quad (a,m)=1$$

有解，则称 a 为模 m 的平方剩余（quadratic residue，也叫做二次剩余），否则称 a 为模 m 的平方非剩余（quadratic non-residue，二次非剩余）。

思考 4.1：对于 $m=2$，判断某个数是否为模 2 的平方剩余是平凡的。

下面主要考虑模为奇素数 p 的平方剩余。

例 4.1　求模 7 的平方剩余。

解答：通过穷举法，可计算出

$$1^1 \equiv 1 \pmod 7 \quad 2^2 \equiv 4 \pmod 7 \quad 3^2 \equiv 2 \pmod 7$$
$$4^2 \equiv 2 \pmod 7 \quad 5^2 \equiv 4 \pmod 7 \quad 6^2 \equiv 1 \pmod 7$$

于是,1,2,4 是模 7 的平方剩余,3,5,6 是模 7 的平方非剩余。

通过观察,发现在计算时可以“成对”计算,即只需要计算一半即$(p-1)/2$个数即可:

$$1^1 \equiv 1 \pmod 7 \qquad 2^2 \equiv 4 \pmod 7 \qquad 3^2 \equiv 2 \pmod 7$$
$$6^2 \equiv (-1)^2 \equiv 1 \pmod 7 \quad 5^2 \equiv (-2)^2 \equiv 4 \pmod 7 \quad 4^2 \equiv (-3)^2 \equiv 2 \pmod 7$$

例 4.2 求模 17 的平方剩余。

解答:

x	1,16	2,15	3,14	4,13	5,12	6,11	7,10	8,9
$a \equiv x^2 \pmod{17}$	1	4	9	16	8	2	15	13

模 17 的平方剩余为 1,2,4,8,9,13,15,16;平方非剩余为 3,5,6,7,10,11,12,14。

通过观察,可以发现,平方剩余的个数是简化剩余系元素个数的一半。

一般地,有如下结论:

定理 4.1 在素数模 p 的一个简化剩余系中,恰有$(p-1)/2$个模 p 的平方剩余,$(p-1)/2$个平方非剩余。

在给出证明之前,先看定理示意图(见图 4.1),左边为简化剩余系,右边为平方。可以看到,由于简化剩余系“成对”计算,计算出的平方剩余个数刚好为简化剩余系元素个数的一半。

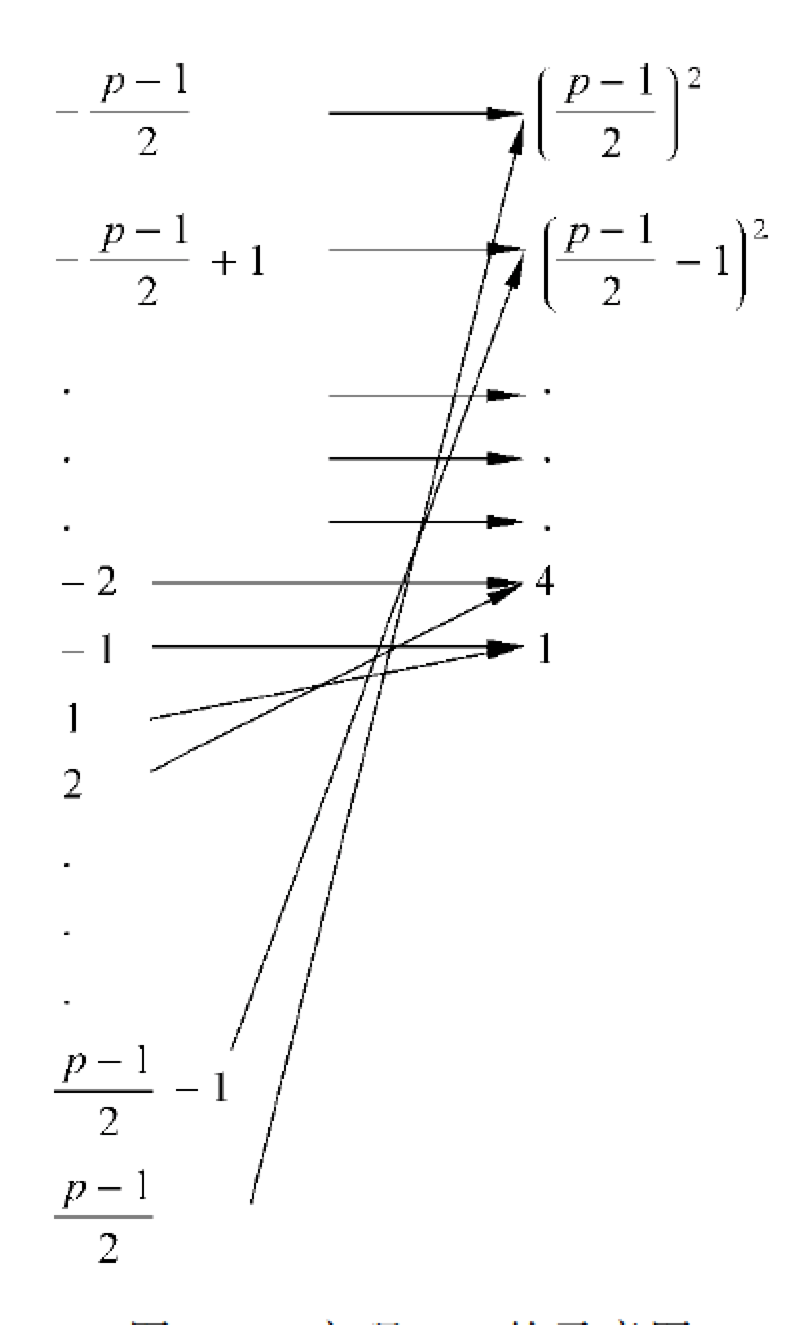


图 4.1 定理 4.1 的示意图

证明: 取模 p 的最小简化剩余系,这里在成对放置

$$-\frac{p-1}{2},-\frac{p-1}{2}+1,\cdots,-2,-1$$

$$\frac{p-1}{2},\frac{p-1}{2}-1,\cdots,+2,+1$$

则 a 是模 p 的平方剩余当且仅当

$$a\equiv\left(-\frac{p-1}{2}\right)^2,\left(-\frac{p-1}{2}+1\right)^2,\cdots,-2,-1,$$
$$\left(\frac{p-1}{2}\right)^2,\left(\frac{p-1}{2}-1\right)^2,\cdots,2,1,\ (\mathrm{mod}\ p)$$

由于$(-i)^2\equiv i^2(\mathrm{mod}\ p)$，所以 a 模 p 的平方剩余当且仅当

$$a\equiv1^2,2^2,\cdots,\left(-\frac{p-1}{2}-1\right)^2,\left(-\frac{p-1}{2}\right)^2(\mathrm{mod}\ p)$$

又当 $i\neq j,1\leqslant i,j\leqslant(p-1)/2$ 时，$i^2\not\equiv j^2(\mathrm{mod}\ p)$，所以模 p 的平方剩余个数为$(p-1)/2$，模 p 的平方非剩余的个数为 $p-1-(p-1)/2=(p-1)/2$。 ■

根据定理 4.1，可以给出一个算法输出奇素数 p 的平方剩余和平方非剩余。

思考 4.2：如何根据定理 4.1 给出一个算法计算输出奇素数 p 的平方剩余和平方非剩余。

例 4.3　求方程 $E:y^2\equiv x^3+x+2\ (\mathrm{mod}\ 7)$ 的所有点。

解答：方程 E 其实就是一个椭圆曲线方程。由于模为 7，对 $x=0,1,2,3,4,5,6$，分别求 y。

$x=0,y^2\equiv2\ (\mathrm{mod}\ 7)$，$y=3,4\ (\mathrm{mod}\ 7)$

$x=1,y^2\equiv4\ (\mathrm{mod}\ 7)$，$y=2,5\ (\mathrm{mod}\ 7)$

$x=2,y^2\equiv5\ (\mathrm{mod}\ 7)$，无解

$x=3,y^2\equiv4\ (\mathrm{mod}\ 7)$，$y=2,5\ (\mathrm{mod}\ 7)$

$x=4,y^2\equiv0\ (\mathrm{mod}\ 7)$，$y=0\ (\mathrm{mod}\ 7)$

$x=5,y^2\equiv6\ (\mathrm{mod}\ 7)$，无解

$x=6,y^2\equiv0\ (\mathrm{mod}\ 7)$，$y=0\ (\mathrm{mod}\ 7)$

根据该结果，可以画出椭圆曲线图（见图 4.2），这个看上去像个"围棋盘"的图中，如果 $y\neq0$，则 y 坐标是关于 7/2 对称的。（关于椭圆曲线密码的详述，请见第 9 章。）

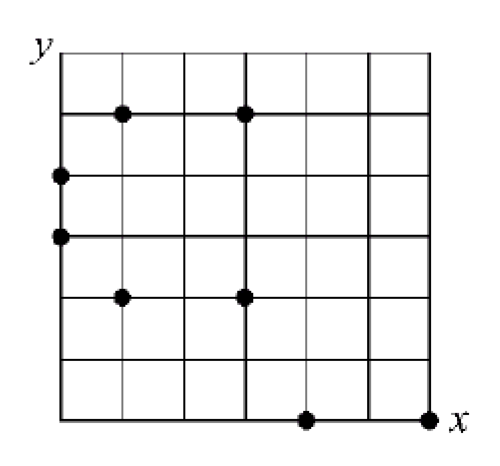


图 4.2　椭圆曲线方程 $E:y^2\equiv x^3+x+2\ (\mathrm{mod}\ 7)$上所有的点

前面（例 4.1 和例 4.2）对平方剩余的判定主要依靠穷举法，下面给出欧拉判定法则，从理论上给出了判别 a 是否为模 p 的平方剩余的方法。该方法将平方剩余的判定问题转化成计算问题。

定理 4.2(Euler 判定法则) 设 p 是奇素数,$(a,p)=1$,则

(1) a 是模 p 的平方剩余的充要条件是

$$a^{\frac{p-1}{2}}\equiv 1 \pmod p$$

(2) a 是模 p 的非平方剩余的充要条件是

$$a^{\frac{p-1}{2}}\equiv -1 \pmod p$$

当且仅当 a 是模 p 的平方剩余时,二次同余式

$$x^2\equiv a \pmod p,\ (a,p)=1$$

有 2 解。

证明: 先证明(1)。先证明必要性。假定 $a\equiv y^2 \bmod p, a>0$,故 $y\not\equiv 0 \bmod p$。于是根据 Fermat 定理,$(y^2)^{(p-1)/2}\equiv y^{p-1}\equiv 1 \bmod p$。

再证充分性。假定 $a^{(p-1)/2}\equiv 1 \bmod p$,设 b 为一个模 p 的原根(见第 5 章),于是 $a\equiv b^i \bmod p$,对于某个正整数 i 成立。有 $a^{(p-1)/2}\equiv b^{i(p-1)/2}\equiv b^{i(p-1)/2} \bmod p$,由于 b 的阶(见第 5 章)为 $p-1$,因此必有 $(p-1)\mid(i(p-1)/2)$。因此,i 是偶数,于是 a 的平方根为 $\pm b^{i/2} \bmod p$。

下面证明(2)。由于 $a^{p-1}\equiv 1 \pmod p$。$a^{\frac{p-1}{2}}\equiv 1 \pmod p$ 或者 $a^{\frac{p-1}{2}}\equiv -1 \pmod p$。由(1)可知(2)成立。 ■

例 4.4 8 是不是模 17 的平方剩余?

解答: $8^{(17-1)/2}\equiv 8^8\equiv 1 \pmod{17}$,因此,8 是模 17 的平方剩余。

例 4.5 137 是不是模 227 的平方剩余?

解答: 计算 $137^{(227-1)/2}\equiv 137^{113} \pmod{227}$,利用模重复平方法,得到该值为 -1,因此,137 是模 227 平方非剩余。

推论: 设 p 是奇素数,$(a_1,p)=1,(a_2,p)=1$,则

(1) 如果 a_1,a_2 都是模 p 的平方剩余,则 a_1a_2 是模 p 的平方剩余;

(2) 如果 a_1,a_2 都是模 p 的平方非剩余,则 a_1a_2 是模 p 的平方非剩余;

(3) 如果 a_1 是模 p 的平方剩余,a_2 是模 p 的平方非剩余,则 a_1a_2 是模 p 的平方非剩余。

显然,定理 4.2 可以转换成一个算法。虽然该算法比较简单,但是有利于加深对概念的理解。

算法 4.1 Euler 判定方法计算 a 是否为模 p 的平方剩余。

```
/* int Euler(a, p)                                  */
/* 输入:整数 a, (a,p)=1,奇素数 p                   */
/* 输出:a是平方剩余或者 a是平方非剩余              */
{
int ex = Square-and-Multiply(a,(p-1)/2,p); //调用模重复平方子函数
if (ex==1)
{
        printf("a是平方剩余");
```

```
        Return 1;
    }
    else
    {
        printf("a是平方非剩余");
        Return - 1;
    }
}
```

4.2 Legendre 符号及其计算方法

当 p 不太大时，可以利用算法 4.1 判定某个数是否为平方剩余。但是，当 p 比较大时，该方法的计算量较大，就不实用了。下面引入勒让德符号，给出一种判定模 p 平方剩余的更有效的方法。

定义 4.2　设 p 是素数，a 是整数，Legendre(勒让德)符号定义如下：

$$\left(\frac{a}{p}\right)=\begin{cases}1, & \text{若 } a \text{ 是模 } p \text{ 的平方剩余}\\ -1, & \text{若 } a \text{ 是模 } p \text{ 的平方非剩余}\\ 0, & \text{若 } p \mid a\end{cases}$$

Legendre(勒让德)符号可理解成一个判定函数，函数的返回值为 1，−1，0。

算法 4.2 Legendre 函数计算 Legendre 符号的结果，这一计算过程直接从定义给出，是平凡的，主要目的是帮助初学者加深对 Legendre 符号这一概念的理解。

算法 4.2　Legendre 函数计算 Legendre 符号的结果。

```
/* int Legendre(int a, int p)        */
/* 输入：整数 a，素数 p              */
/* 输出：Legendre 符号的值           */
int Legendre(int a, int p)
{
    if (p% a==0)                //p|a
        Return 0;
    else
    {
        if (Euler(a,p)==1)   //调用 Euler 判定函数
            Return 1;
        else
            Return -1;
    }
}
```

其实，如果不要求定理 4.2 中的 $(a,p)=1$，并且将 Legendre 符号引入，定理 4.2 可以叙述为定理 4.3。

定理 4.3 （欧拉判定法则）设 p 是奇素数，则对任意整数 a，

$$\left(\frac{a}{p}\right)=a^{\frac{p-1}{2}}\ (\bmod\ p)$$

定理 4.3 有两个好处：将 Legendre 符号的计算转变成模幂计算问题；同时这一计算结果可用来判断 a 是否为模 p 的平方剩余。

算法 4.3 计算 Legendre 符号（由定义给出，通过模幂计算的平凡方法）。

```
/* int L (int a, int p)            */
/* 输入：整数 a, 奇素数 p          */
/* 输出：Legendre 符号的值         */
int L(int a, int p)
{
    Return Square-and-Multiply(a,(p-1)/2,p);  //调用模重复平方函数
}
```

下面讨论如何给出 Legendre 符号的快速计算方法。

由定理 4.3，可得到如下推论（这些推论有利于 Legendre 符号的快速计算）。

推论 1 设 p 是奇素数，则

(1) $\left(\frac{1}{p}\right)=1$。

(2) $\left(\frac{-1}{p}\right)=(-1)^{\frac{p-1}{2}}$。

由推论 1 可以给出如下计算子函数的定义，帮助初学者理解。

```
int L_ONE(const int a=1, int p)
{
    Retrun 1;
}
```

```
int L_MinusOne(const int a=-1, int p)
{
    Retrun (-1)^((p-1)/2);
}
```

推论 2 设 p 是奇素数，那么

$$\left(\frac{-1}{p}\right)=\begin{cases}1, & \text{若 } p\equiv 1\ (\bmod\ 4)\\ -1, & \text{若 } p\equiv 3\ (\bmod\ 4)\end{cases}$$

证明：根据 Euler 判别法则，有

$$\left(\frac{-1}{p}\right)=(-1)^{\frac{p-1}{2}}$$

若 $p\equiv1\pmod 4$，则存在正整数 k，使得 $p=4k+1$，于是 $(p-1)/2=2k$。从而

$$\left(\frac{-1}{p}\right)=(-1)^{2k}=(-1)^{2k}=1$$

若 $p\equiv1\pmod 4$，则存在正整数 k，使得 $p=4k+1$，于是 $(p-1)/2=2k+1$。从而

$$\left(\frac{-1}{p}\right)=(-1)^{2k+1}=(-1)^{2k+1}=-1$$

■

定理 4.4　设 p 是奇素数，则

(1) $\left(\frac{a}{p}\right)=\left(\frac{a+p}{p}\right)$。

(2) $\left(\frac{ab}{p}\right)=\left(\frac{a}{p}\right)\left(\frac{b}{p}\right)$。

(3) 若 $(a,p)=1$，则 $\left(\frac{a^2}{p}\right)=1$。

思考 4.3　读者能否给出一些例子说明上述定理。

定理 4.5　设 p 是奇素数，则

$$\left(\frac{2}{p}\right)=(-1)^{\frac{p2-1}{8}}$$

例 4.6　证明 2 是模 17 的平方剩余。

证明：根据定理 4.5，

$$\left(\frac{2}{17}\right)=(-1)^{\frac{17^2-1}{8}}=(-1)^{2\cdot18}=1$$

因此，2 是模 17 的平方剩余。■

定理 4.5 可以得到如下子函数（比较简单，仅为加深理解）。

```
int L_TWO(const int a=2, int p)
{
    Retrun (-1)^((p^2-1)/8);
}
```

定理 4.6（二次互反律）　设 p，q 是互素的奇素数，则

$$\left(\frac{p}{q}\right)=(-1)^{\frac{p-1}{2}\frac{q-1}{2}}\left(\frac{q}{p}\right)$$

写成一个子函数，可以得到：

```
int L_MutualInverse(p,q)
{
    Retrun (-1)^((p-1)/2·(q-1)/2) L(q,p);
}
```

二次互反律提供了 Lengendre 符号的递归计算方式。

例 4.7 证明 3 是模 17 的平方非剩余。

证明：

$$\left(\frac{3}{17}\right)=(-1)^{\frac{3-1}{2}\cdot\frac{17-1}{2}}\left(\frac{17}{3}\right)=\left(\frac{-1}{3}\right)=(-1)^{\frac{3-1}{2}}=-1$$

因此，3 是模 17 的平方非剩余。 ■

例 4.8 判断同余式 $x^2\equiv 137 \pmod{227}$ 是否有解。

解答：因为 227 是奇素数，根据定理 4.4，

$$\left(\frac{137}{227}\right)=\left(\frac{-90}{227}\right)=\left(\frac{-1}{227}\right)\left(\frac{2\cdot 3^2\cdot 5}{227}\right)=-\left(\frac{2}{227}\right)\left(\frac{5}{227}\right)$$

$$\left(\frac{2}{227}\right)=(-1)^{\frac{227^2-1}{8}}=(-1)^{\frac{226\cdot 228}{8}}=-1$$

$$\left(\frac{5}{227}\right)=(-1)^{\frac{5-1}{2}\frac{227-1}{2}}\left(\frac{227}{5}\right)=\left(\frac{2}{5}\right)=(-1)^{\frac{5^2-1}{8}}=-1$$

因此，$\left(\frac{137}{227}\right)=-1$。

故同余式 $x^2\equiv 137 \pmod{227}$无解。

由上述的定理和推论，可以得到如下计算 Legendre 符号的算法 4.4。

算法 4.4 Legendre(勒让德)符号的计算方法。

/* int L_FAST(int a, int p)

/* 输入：奇素数 p≥3，整数 a，且 0≤a< p */

/* 输出：勒让德符号 $\left(\frac{a}{p}\right)$ */

1. 如果 a=0，则返回 0;
2. 如果 a=1，则返回 1;
3. 令 $a=2^e a1$，其中 a1 为奇数;
4. 如果 e 是偶数，则令 s=1;否则，如果 p=1 或 7 (mod 8)，则令 s←1;如果 p=3 或者 5 (mod 8)，则令 s←-1;
5. 如果 p=3 (mod 4)，a1=3(mod 4)，则令 s←-s;
6. 令 p1 ← p mod a1;
7. 如果 a1=1，则返回 s;否则返回 s·L_FAST(p1,a1)。

例 4.9 判断同余式

$$x^2\equiv -1 \pmod{365}$$

解答：365 可分解成 2 个素数的乘积，即 365=5·73，原同余式等价于

$$\begin{cases}x^2\equiv -1 \pmod 5\\ x^2\equiv -1 \pmod{73}\end{cases}$$

因为

$$\left(\frac{-1}{5}\right)=\left(\frac{-1}{73}\right)=1$$

故同余式组有解，根据 CRT，同余式组中的每个同余式均有 2 个解，2 个同余式的 2 个解分别配对，于是产生 4 个同余式组，4 个同余式组的解即为原同余式的解。因此，原同余式的解数为 4。

4.3 Rabin 公钥密码系统

前面 4.1 节和 4.2 节讨论了二次同余式以及平方剩余的判定，它们在密码学中的一个典型应用就是 Rabin 公钥密码系统①。

该公钥密码体制基于一个困难问题，称为平方根问题。

定义 4.3　平方根问题（SQROOT 问题）：给定一个合数 n 和 $a \in Q_n$（模 n 的平方剩余集合），找 a 模 n 的平方根；即找到一个整数 x，使得 $x^2 = a \bmod n$。

但是，如果知道 n 的因子 p 和 q，那么 SQROOT 问题就很容易求解。方法是先求解 a 模 p 和模 q 的解，然后利用中国剩余定理得到 a 模 n 的平方根。

于是，可考虑将 n 作为公钥，将 p 和 q 为私钥。

图 4.3 给出了这一困难问题的示意图。

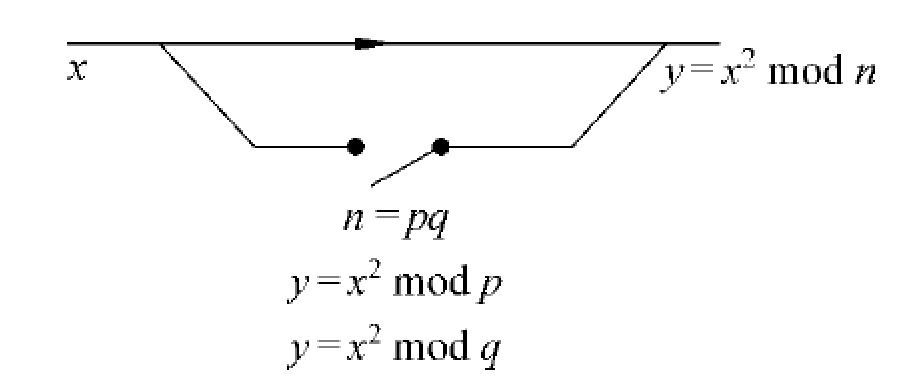


图 4.3　Rabin 利用单向陷门函数的原理示意图

Rabin 公钥密码系统的描述：

（1）密钥的产生：随机生成两个大的素数 p 和 q，满足 $p \equiv q \equiv 3 \bmod 4$，计算 $n = pq$。n 为公钥，p，q 作为私钥。

（2）加密：$c = m^2 \bmod n$。

（3）解密：解密就是求 c 模 n 的平方根，即解 $x^2 = c \bmod n$，该方程等价于方程组

$$\begin{cases} x^2 \equiv c \pmod{p} \\ x^2 \equiv c \pmod{q} \end{cases}$$

由于 $p \equiv q \equiv 3 \bmod 4$，可容易地求出 c 在模 p 下的 2 个方程根

$$m \equiv c^{(p+1)/4} \bmod p$$

$$m \equiv p - c^{(p+1)/4} \bmod p$$

和 c 在模 q 下的 2 个方程根：

① Rabin 公钥密码体制是 1979 年由 MIT 的 M. O. Rabin 在其论文"Digitalized Signatures and Public-Key Functions as Intractable as Factorization"中提出。

$$m \equiv c^{(q+1)/4} \bmod q$$
$$m \equiv q - c^{(q+1)/4} \bmod q$$

两两组合联立，可得 4 个方程组：

$$\begin{cases} m \equiv c^{(p+1)/4} \bmod p \\ m \equiv c^{(q+1)/4} \bmod q \end{cases}$$
$$\begin{cases} m \equiv p - c^{(p+1)/4} \bmod p \\ m \equiv c^{(q+1)/4} \bmod q \end{cases}$$
$$\begin{cases} m \equiv c^{(p+1)/4} \bmod p \\ m \equiv q - c^{(q+1)/4} \bmod q \end{cases}$$
$$\begin{cases} m \equiv p - c^{(p+1)/4} \bmod p \\ m \equiv q - c^{(q+1)/4} \bmod q \end{cases}$$

利用中国剩余定理求得 4 个 m，其中必有一个 m 为明文。这通常可以通过在明文中事先引入某些特征（如发送者的身份号、日期等事先约定的特征）来区分。

对该密码方案的解释：

思考 4.4　为什么取 $p \equiv q \equiv 3 \bmod 4$。

目的是为了能快速的求出平方根。$p \equiv 3 \bmod 4$ 时，$x^2 \equiv c \bmod p$ 的平方根容易求出。

由 $p \equiv 3 \bmod 4$，得 $p+1=4k$，即 $(p+1)/4$ 是一个整数。设 $x^2 \equiv c \bmod p$ 的根为 y，即 $y^2 \equiv c \bmod p$。

因 c 是模 p 的二次剩余，故 $c^{(p-1)/2} \equiv (y^2)^{(p-1)/2} \equiv y^{p-1} \equiv 1 \bmod p$。

于是，

$$(c^{\frac{p+1}{4}})^2 \equiv (y^{\frac{p+1}{2}})^2 \equiv c^{\frac{p+1}{2}} \equiv c^{(p-1)/2} \cdot c \equiv c \bmod p$$

故 $c^{(p+1)/4}$ 和 $p - c^{(p+1)/4}$ 是方程 $x^2 \equiv c \bmod p$ 的两个根。同理，$c^{(q+1)/4}$ 和 $q - c^{(q+1)/4}$ 是方程 $x^2 \equiv c \bmod q$ 的两个根。

思考 4.5　为什么解 $x^2 \equiv c \bmod n$ 方程等价于解方程组 $\begin{cases} x^2 \equiv c \pmod p \\ x^2 \equiv c \pmod q \end{cases}$。

下面证明一个一般的结论：当 p,q 是素数，$n=pq$ 时，$a \equiv b \bmod n$ 等价于方程组

$$\begin{cases} a \equiv b \pmod p \\ a \equiv b \pmod q \end{cases}$$

证明：若 $\begin{cases} a \equiv b \pmod p \\ a \equiv b \pmod q \end{cases}$，则 $p \mid (a-b)$，$q \mid (a-b)$，而 $\gcd(p,q)=1$，所以 $pq \mid (a-b)$，即 $a \equiv b \bmod pq$。

反之，若 $a \equiv b \bmod n$，则 $n \mid (a-b)$，由 $p \mid n$，$q \mid n$，得 $p \mid (a-b)$，$q \mid (a-b)$，即

$$\begin{cases} a \equiv b \pmod p \\ a \equiv b \pmod q \end{cases}$$ ■

例 4.10　Rabin 方案的举例。假设私钥为 $p=7$，$q=11$（p,q 模 4 余 3），公钥为 $n=pq=77$，明文 $m=32$，则密文为 $c=m^2 \bmod n=32^2 \bmod 77=23$。下面分析解密密文 23 的

过程。$m \equiv \sqrt{c} \bmod 77$。分别求模 p,q 的平方根。

$$23^{7+1/4} \equiv 2^2 \equiv 4 \bmod 7$$

$$23^{11+1/4} \equiv 1^2 \equiv 1 \bmod 11$$

23 模 7 和模 11 的平方根是 ± 4 和 ± 1，然后利用中国剩余定理，计算得到 23 模 77 的 4 个平方根，± 10，$\pm 32 \pmod{77}$，4 个可能的明文 10，32，45，67。在实际中，通过预先在明文中加入识别特征（例如约定添加特殊标识符号到明文的末尾，或者将明文重复 1 次再连接后作为新的明文，即 $m' = m||m$），可以从 4 个可能的明文中选出正确的明文。

由于 Rabin 加密算法十分简单，下面重点解释解密算法。

定理 4.7　设 p,q 是形为 $4k+3$ 的不同素数，如果整数 a 满足

$$\left(\frac{a}{p}\right) = \left(\frac{a}{q}\right) = 1$$

则同余式 $x^2 \equiv a \pmod{pq}$ 有解

$$x = \pm (a^{(p+1)/4} (\bmod p)) \cdot (q^{-1} (\bmod p)) \cdot q \pm (a^{(q+1)/4} (\bmod q)) \cdot (p^{-1} (\bmod q)) \cdot p \pmod{pq}$$

证明： 二次同余式 $x^2 \equiv a \pmod{pq}$ 等价于同余式组

$$\begin{cases} x^2 \equiv a \pmod p \\ x^2 \equiv a \pmod q \end{cases}$$

因为 $\left(\frac{a}{p}\right) = \left(\frac{a}{q}\right) = 1$，所以由思考 4.5，同余式 $x^2 \equiv a \pmod p$ 有解，其解为

$$x = \pm a^{(p+1)/4} (\bmod p)$$

同样地，同余式 $x^2 \equiv a \pmod q$ 有解，其解为

$$x = \pm a^{(q+1)/4} (\bmod q)$$

根据中国剩余定理，同余式组的解为

$$x = \pm (a^{(p+1)/4} (\bmod p)) \cdot (q^{-1} (\bmod p)) \cdot q \pm (a^{(q+1)/4} (\bmod q)) \cdot (p^{-1} (\bmod q)) \cdot p \pmod{pq}$$

这即为同余式 $x^2 \equiv a \pmod{pq}$ 的解。■

根据定理 4.6 给出 Rabin 解密的算法。

算法 4.5　Rabin 解密算法。

```
/* 输入：密文 c，私钥 p,q(p,q均为 4k+3的素数，n=pq)  */
/* 输出：明文 m                                      */
RabinDecrypt(c,p,q){
1. 利用扩展的欧几里得除法，求整数 s，t，使得 sp+tq=1
2. 计算 u=c^((p+1)/4) mod p
3. 计算 v=c^((q+1)/4) mod q
4. 计算 m1=(utq+vsp) mod n //t 即定理 4.6中 q^-1(mod p)，s 即 p^-1(mod q)
5. 计算 m2=(utq-vsp) mod n
6. 同余式 m^2≡ c (mod n) 的 4个根是 m1，n-m1,m2，n-m2，确定其中哪一个根为明文。
}
```

思考 4.6 Rabin 加密可否视为 RSA 加密的特例？

其实，Rabin 加密不是 RSA 加密的特例。因为 RSA 加密中的私钥 e 必须有 $\gcd(e,\Phi(n))=1$，而 $\Phi(n)=(p-1)(q-1)$ 为偶数，故 e 必为奇数。但是，Rabin 加密中使用的幂次是 2，为偶数。

思　考　题

[1] 求模 $p=13, 23, 31, 37, 47$ 的平方剩余和平方非剩余。

[2] 求满足方程 $E: y^2=x^3-3x+1 \pmod 7$ 的所有点。

[3] 计算 Legendre 符号 $\left(\frac{17}{37}\right)$，$\left(\frac{151}{373}\right)$，$\left(\frac{191}{397}\right)$，$\left(\frac{911}{2003}\right)$。

[4] 编写程序 Legendre 符号，验证题目[3]的结果。

[5] 同余式是否有解 $x^2\equiv 7 \pmod{227}$，$x^2\equiv 2 \pmod{401\cdot 281}$。

[6] 编写程序实现 300 十进制位的 Rabin 公钥密码系统。

第5章 原根与指数

在研究了2次剩余之后，下面讨论 n 次剩余。

讨论使得同余式

$$a^n \equiv 1 \pmod{m}$$

成立的整数 n。

这里主要关心最小的正整数 e，因为找到了 e，则 e 的倍数均为上式的解。另外，由 Euler 定理可知：当 $(a,m)=1$ 时，有 $a^{\varphi(m)} \equiv 1 \pmod{m}$。于是，值得关心的一个问题，就是这个最小的正整数 e 会不会就是 $\varphi(m)$，什么情况下就是 $\varphi(m)$。

本章的重点是原根、阶及其计算方法。难点是 DH 密钥协商和 ElGamal 公钥密码系统。

5.1 原根和阶的概念

定义 5.1 设 $m>1$ 是整数，a 是正整数，$(a,m)=1$，则使得

$$a^x \equiv 1 \pmod{m}$$

成立的最小正整数 x 叫作 a 模 m 的阶(Order)。记为 $\mathrm{ord}_m(a)$。

例 5.1 设整数 $m=7$，计算 $a=1,2,3,4,5,6$ 的阶。

解答： $1^1 \equiv 1 \pmod{7}$　　$2^3 \equiv 1 \pmod{7}$　　$3^3 \equiv -1 \pmod{7}$

$4^3 \equiv (-3)^3 \equiv 1 \pmod{7}$　　$5^3 \equiv (-2)^3 \equiv -1 \pmod{7}$　$6^2 \equiv (-1)^2 \equiv 1 \pmod{7}$

a	1	2	3	4	5	6
$\mathrm{ord}_m(a)$	1	3	6	3	6	2

容易看到，由于 $\varphi(m) = \varphi(7) = 6$，于是，$\mathrm{ord}_m(a)$ 中最大为 6。

定义 5.2 原根(Primitive Root)。如果 $\mathrm{ord}_m(a)=\varphi(m)$，则 a 叫做 m 的原根。

思考 5.1： 原根的英文可能初学者觉得很难理解，如果称为“生成元(generator)”可能更容易理解这个“本原(Primitive)”的含义，即可以生成群中其他所有元素的“本原”元。

容易看到，在例 5.1 中，因为 3 和 5 的阶为 $\varphi(7)$，所以 3 和 5 为原根。

从生成元的角度更加容易理解，但需要群的概念。群 $(Z/7Z)^*$ 中元素的个数为 $\varphi(7)=6$，因为 3 和 5 的指数为 $\varphi(7)$，故 3 和 5 可以生成群 $(Z/7Z)^*$ 中的任意元素(即 1～6)。

首先考查3：

$3^1 \equiv 3 \pmod 7$　　$3^2 \equiv 2 \pmod 7$　　$3^3 \equiv 6 \pmod 7$

$3^4 \equiv 4 \pmod 7$　　$3^5 \equiv 5 \pmod 7$　　$3^6 \equiv 1 \pmod 7$

容易看到，在图5.1中，3可以生成群中的元素1～6。

再来考查5：

$5^1 \equiv 5 \pmod 7$　　$5^2 \equiv 4 \pmod 7$　　$5^3 \equiv 6 \pmod 7$

$5^4 \equiv 2 \pmod 7$　　$5^5 \equiv 3 \pmod 7$　　$5^6 \equiv 1 \pmod 7$

除了3和5外，其他都不是原根，因为其指数均小于$\varphi(7)$。即无法生成“所有”元素，“回到原点”1的过程过快，导致只能生成群中的“部分”元素。

考查2：

$2^1 \equiv 2 \pmod 7$　　$2^2 \equiv 4 \pmod 7$　　$2^3 \equiv 1 \pmod 7$

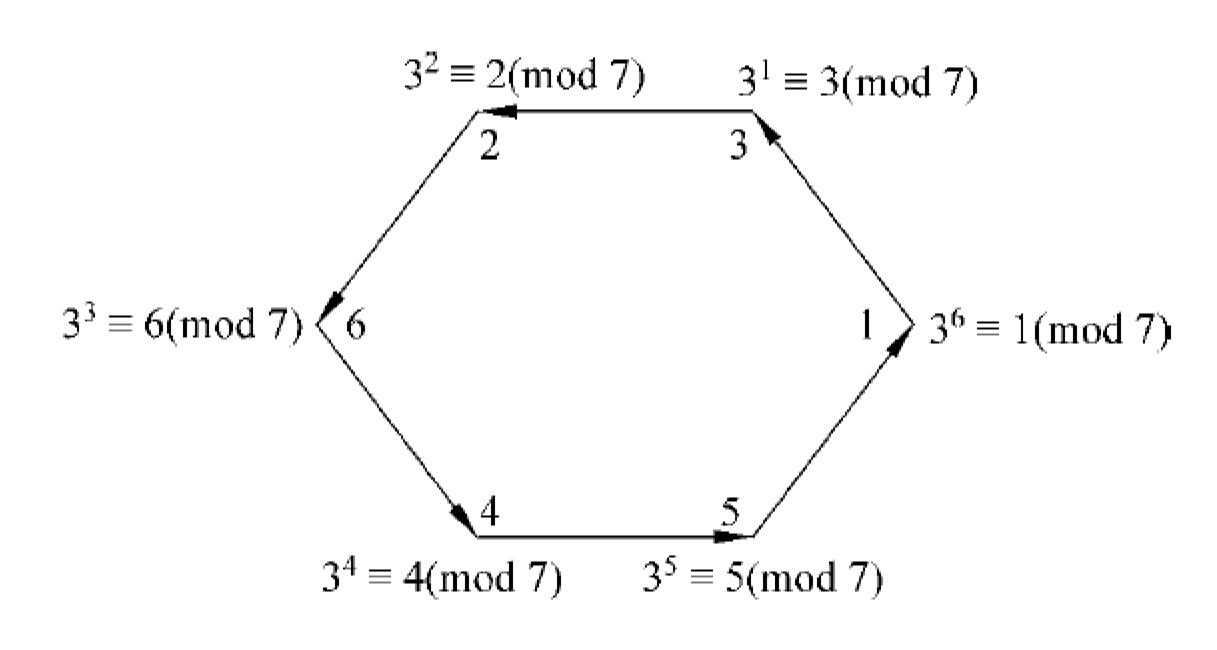


图5.1　3作为生成元生成$(Z/7Z)^*$中所有元素

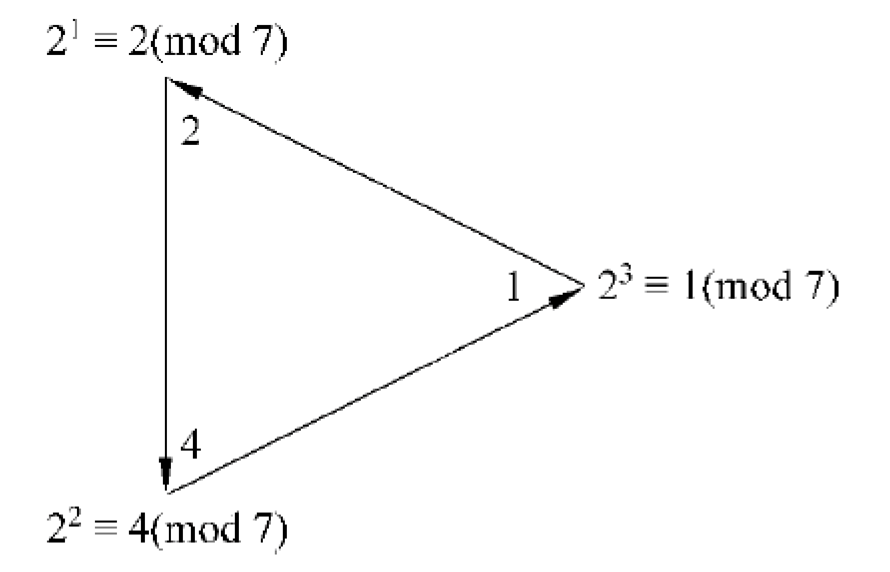


图5.2　2只能生成$(Z/7Z)^*$中3个元素

容易看到，在图5.2中，2无法生成3，5，6。

从上述两个图示，就容易理解为什么原根的阶为$\varphi(m)$了。

下面看一个m不为素数的例子。

例5.2　设整数$m=14$，计算$a=1,3,5,7,9,11,13$的阶，指出其中的原根。

解答： $1^1 \equiv 1 \pmod{14}$　　$3^3 \equiv -1 \pmod{14}$　　$5^3 \equiv -1 \pmod{14}$

$9^3 \equiv 1 \pmod{14}$　　$11^3 \equiv 1 \pmod{14}$　　$13^2 \equiv 1 \pmod{14}$

a	1	3	5	9	11	13
$\mathrm{ord}_m(a)$	1	6	6	3	3	2

由于$\varphi(14)=\varphi(2)\cdot\varphi(7)=\varphi(7)=6$，因此原根为3和5。

下面的定理给出了计算阶的一个算法依据。

定理5.1　设$m>1$是整数，a为整数，$(a,m)=1$，则整数d使得

$$a^d \equiv 1 \pmod m$$

成立的充要条件是

$$\mathrm{ord}_m(a) \mid d$$

证明： 先证明充分性。如果$\mathrm{ord}_m(a) \mid d$，那么存在整数k，使得$d=k\cdot\mathrm{ord}_m(a)$。因此，有

$$a^d = (a^{\mathrm{ord}_m(a)})^k \equiv 1 \pmod m$$

再证明必要性。如果 $\mathrm{ord}_m(a)|d$ 不成立,则存在整数 q,r,使得

$$d = \mathrm{ord}_m(a) \cdot q + r,\ 0 < r < \mathrm{ord}_m(a)$$

从而,

$$a^d = a^r (a^{\mathrm{ord}_m(a)})^q \equiv a^r \pmod m$$

又因为

$$a^d \equiv 1 \pmod m$$

于是

$$a^r \equiv 1 \pmod m$$

这与 $\mathrm{ord}_m(a)$ 的最小性矛盾。于是有 $\mathrm{ord}_m(a)|d$。 ■

推论 设 $m>1$ 是整数,a 为整数,$(a,m)=1$,则 $\mathrm{ord}_m(a)|\varphi(m)$。

证明 根据 Euler 定理,有

$$a^{\varphi(m)} \equiv 1 \pmod m$$

由定理 5.1,于是 $\mathrm{ord}_m(a)|\varphi(m)$。 ■

由上述推论可知,可以从 $\varphi(m)$ 中寻找 $\mathrm{ord}_m(a)$。

例 5.3 求整数 5 模 17 的阶 $\mathrm{ord}_{17}(5)$。

解答:因为 $\varphi(17)=16$,只需要对 16 的因数 $d=1,2,4,8,16$ 计算 $5^d \pmod{17}$,看是否等于 1。因为

$5^1 \equiv 5 \pmod{17}$　　$5^2 \equiv 8 \pmod{17}$　　$5^4 \equiv 64 \equiv 13 \equiv -4 \pmod{17}$

$5^8 \equiv (-4)^2 \equiv 16 \equiv -1 \pmod{17}$　　$5^{16} \equiv (-1)^2 \equiv 1 \pmod{17}$

所以 5 是模 17 的原根。

有了原根的概念后,可以给出指数(Index)的概念。

定义 5.3 设 g 是正整数 m 的原根,若 $\gcd(a,m)=1$,则称同余式

$$g^x \equiv a \pmod m$$

的唯一整数解 $x(1 \leqslant x \leqslant \Phi(m))$ 为 a 模 m 以 g 为底的指数(Index),也称为指标或离散对数,记为 $\mathrm{Ind}_{g,m}(a)$,或简记为 $\mathrm{Ind}_g a$。①

离散对数问题是一个计算上的困难的问题,目前还没有找到有效的算法。5.3 节和 5.4 节会给出基于该困难问题构造的密码系统。第 11 章会讲解离散对数算法。

例 5.4 设 $m=7$,由例 5.1 可知,3 为原根。计算指数表。

解答:

a	1	2	3	4	5	6
$\mathrm{Ind}_3 a$	6	2	1	4	5	3

思考 5.2:"阶"和"指数"的主要区别是什么?

容易看到,阶与模 m 和整数 a 有关,指数则与模 m,整数 a 以及底(原根 g)有关。阶

① 有些书籍在英译方面把 ord 译为"指数",而用"指标"表示离散对数。本书倾向于 ord 译为"阶"的缩写,"指数"和"指标"与离散对数等同,缩写为 ind。

可以用来判断原根，指数主要用来计算元素相对于原根的离散对数。

定理 5.2（指数定理） 若 g 是模 m 的一个原根，则 $g^x \equiv g^y \pmod{m}$ 当且仅当 $x \equiv y \pmod{\varphi(m)}$。

证明： 假设 $x \equiv y \pmod{\varphi(m)}$，则 $x = y + k\varphi(m)$，$k \in Z$，所以

$$g^x \equiv g^{y+k\varphi(m)} \pmod{m} \equiv g^y (g^{\varphi(m)})^k \pmod{m} \equiv g^y 1^k \pmod{m} \equiv g^y \pmod{m}$$

必要性留作练习。 ■

这个证明和 RSA 的解密过程有相似之处。

定理 5.3 设 g 是模素数 p 的一个原根，且 $\gcd(a, p) = 1$，则 $g^k \equiv a \pmod{p}$ 当且仅当

$$k \equiv \text{ind}_g a \pmod{(p-1)}$$

定理 5.4 设 m 是有原根 g 的正整数，a 与 b 是与 m 相互素的整数，则

（1）若 $b \equiv a \pmod{m}$，则 $\text{ind}_g a \equiv \text{ind}_g a$。

（2）$\text{ind}_g 1 \equiv 0 \pmod{\varphi(m)}$。

（3）$\text{ind}_g (a \cdot b) \equiv \text{ind}_g a + \text{ind}_g b \pmod{\varphi(m)}$。

（4）$\text{ind}_g a^k \equiv k \cdot \text{ind}_g a \pmod{\varphi(m)}$，$k$ 是一个正整数。

定义 5.4 设 m 是大于 1 的整数，a 是与 m 互素的整数，如果 n 次同余式

$$x^n \equiv a \pmod{m}$$

有解，则 a 叫做模 m 的 n 次剩余。否则，a 称作模 m 的 n 次非剩余。

定理 5.5 设 m 是大于 1 的整数，g 是模 m 的原根，a 是与 m 互素的整数，则同余式

$$x^n \equiv a \pmod{m}$$

有解的充要条件是

$$(n, \varphi(m)) \mid \text{ind}_g a$$

且在有解的条件下，解数为 $(n, \varphi(m))$。

证明： 由同余式 $x^n \equiv a \pmod{m}$ 和 $(a, m) = 1$，可得 $(x, m) = 1$，于是以 g 为底的 x 的对模 m 的指数存在，设为 y，即 $x \equiv g^y \pmod{m}$，同余式可转化为：

$$g^{ny} \equiv a \pmod{m}$$

由定理 5.5 可知

$$ny \equiv \text{ind}_g a \pmod{\varphi(m)}$$

这是关于 y 的一次同余式，根据定理 3.1，其有解的充要条件是 $(n, \varphi(m)) \mid \text{ind}_g a$，且解数为 $(n, \varphi(m))$。 ■

例 5.5 求解同余式 $4^x \equiv 16 \pmod{17}$ 所有解。

解答： 两边取底为 5 模 17 的指数，得到

$$\text{ind}_5 (4^x) \equiv \text{ind}_5 16 \equiv 8 \pmod{16}$$

即

$$\text{ind}_5 (4^x) \equiv x \cdot \text{ind}_5 4 \equiv 12x \equiv 8 \pmod{16}$$

因此

$$12x \equiv 8 \pmod{16}$$

利用一次同余式的解法（定理 3.1），因为 $(12, 16) = 4$，因此 $12x \equiv 8 \pmod{16}$ 有 4 个

不同余的解，为

$$x \equiv 14,\ 2,\ 6,\ 10 \pmod{16}$$

这即为同余式 $4^x \equiv 16 \pmod{17}$ 的所有解。

5.2 原根与阶的计算

不是所有的模 n 都有原根，下面的定理给出存在原根的条件：

定理 5.6　模 m 的原根存在的充要条件是 $m=2,4,p^\alpha,p^{2\alpha}$，其中 p 是奇素数，α 是一个正整数。

定理 5.7　设 $m>1$，$\varphi(m)$ 的所有不同素因子是 $q_1,q_2,\cdots q_k$，则 g 是模 m 的一个原根的充要条件是

$$g^{\varphi(m)/q_i} \not\equiv 1 \pmod{m},\quad i=1,\cdots,k$$

证明：先证明必要性。设 g 是模 m 的一个原根，则 g 模 m 的阶是 $\phi(m)$。但是

$$0<\varphi(m)/q_i<\varphi(m),\quad i=1,\cdots,k$$

由原根的定义知，

$$g^{\varphi(m)/q_i} \not\equiv 1 \pmod{m},\quad i=1,\cdots,k$$

再证明充分性。

若 g 模 m 的指数 $e<\varphi(m)$，则根据定理 5.1 的推论，有 $e|\varphi(m)$，于是存在一个素数 q，使得 $q \mid \dfrac{\phi(m)}{e}$。于是，根据整除的定义，存在一个整数 u，有

$$qu=\frac{\phi(m)}{e}$$

即

$$eu=\frac{\phi(m)}{q}$$

于是

$$g^{\phi(m)/q}=(g^e)^u \equiv 1 \pmod{m}$$

这与假设矛盾。■

定理 5.8 其实给出了求原根的算法的基础。

例 5.6　求 41 的一个原根。

解答：$\varphi(41)=40=2^3\cdot 5$，素因子为 2 和 5。$\varphi(41)/2=20$，$\varphi(41)/5=8$。因此，只需要验证 g^8，g^{20} 模 m 是否同余于 1。对于 $2,3,\cdots$，逐个验算：

$2^8 \equiv 10 \pmod{41}$　$2^{20} \equiv 1 \pmod{41}$　$3^8 \equiv 1 \pmod{41}$　$4^8 \equiv 18 \pmod{41}$

$4^{20} \equiv 1 \pmod{41}$　$5^8 \equiv 18 \pmod{41}$　$6^8 \equiv 10 \pmod{41}$　$6^{20} \equiv 40 \pmod{41}$

因此，6 是模 41 的原根。

定理 5.8　设 $m>1$ 的整数，a 为整数且 $(a,m)=1$，$d\geqslant 0$ 为整数，则

$$a^d \equiv a^k \pmod{m}$$

的充要条件是

$$d \equiv k \pmod{\mathrm{ord}_m(a)}$$

证明：根据欧几里得除法，存在整数 q, r 和 q', r'，有

$$d = \mathrm{ord}_m(a)q + r, \quad 0 \leqslant r < \mathrm{ord}_m(a)$$
$$k = \mathrm{ord}_m(a)q' + r', \quad 0 \leqslant r' < \mathrm{ord}_m(a)$$

又 $a^{\mathrm{ord}_m(a)} \equiv 1 \pmod m$，于是

$$a^d \equiv a^{\mathrm{ord}_m(a)q} a^r \equiv a^r \pmod m$$
$$a^k \equiv a^{\mathrm{ord}_m(a)q'} a^{r'} \equiv a^{r'} \pmod m$$

必要性得证。若 $a^d \equiv a^k$，则 $a^r \equiv a^{r'} \pmod m$，于是 $r = r'$，有 $d \equiv k \pmod{\mathrm{ord}_m(a)}$

以上步步可逆，知充分性。■

例 5.7 因为整数 2 模 7 的指数为 $\mathrm{ord}_7(2) = 3$，$2015 \equiv 2 \pmod 3$，因此，

$$2^{2015} \equiv 2^2 \pmod 7$$

2.1 节的例 2.1 曾经给出一个实际的应用例子。

定理 5.9 设 $m > 1$ 的整数，a 为整数且 $(a, m) = 1$，$d \geqslant 0$ 为整数，则

$$\mathrm{ord}_m(a^d) = \frac{\mathrm{ord}_m(a)}{(\mathrm{ord}_m(a), d)}$$

思考 5.3：如何利用图 5.1 和图 5.2 给出一个对定理 5.10 的直观解释。

如图 5.3 所示。通俗地说，以原根 3 为度量，$2 = 3^2 \pmod 7$，视为 2"行走"1 步相当于 3"行走"2 步，于是对于 2 而言，"行走"一周 6 步就需要 $6/2 = 3$ 步，即 $\mathrm{ord}_7(2) = 3$。

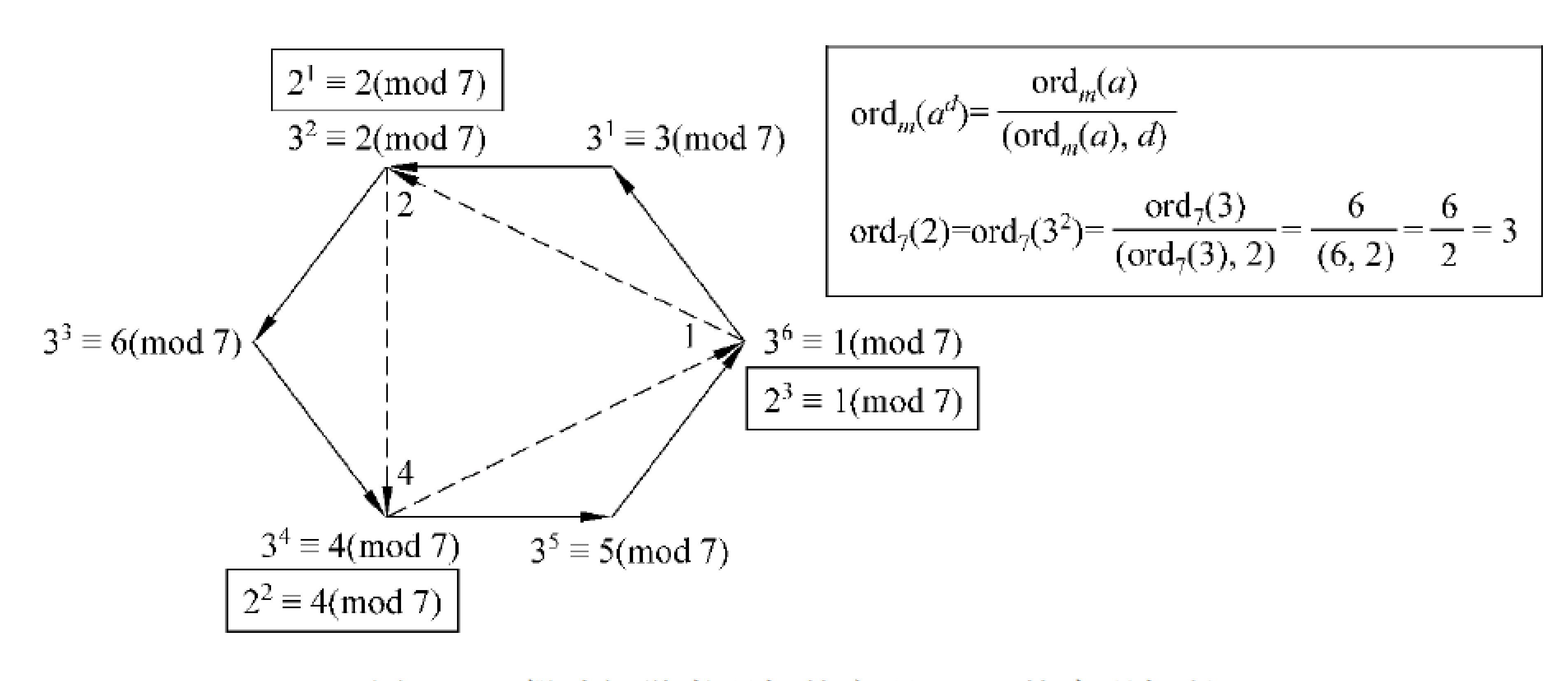


图 5.3　帮助初学者理解的定理 5.10 的直观解释

推论 1：设 $m > 1$ 是整数，g 是模 m 的原根，$d \geqslant 0$ 为整数，则 g^d 是模 m 的原根，当且仅当 $(d, \Phi(m)) = 1$。

推论 2：设 $m > 1$ 是整数，m 有 $\Phi(\Phi(m))$ 个不同的原根。

例如，设素数 $p = 47$，则存在 $\Phi(47 - 1) = 22$ 个模 47 的原根。

目前还没有一种方法可以预知一个给定素数 p 的最小原根，对 $\Phi(p-1)$ 个原根在模 p 的最小剩余系中的分布也知之甚少。

定理 5.9 给出了从一个原根求其他原根的算法基础。

例 5.8 已知 6 是 41 的原根，求 41 的所有原根。

解答：由定理 5.9 知，$(d, \varphi(41)) = 1$ 时，$\mathrm{ord}_{41}(g^d) = \mathrm{ord}_{41}(g)$，因此，当 d 遍历模

$\varphi(41)=40$ 的简化剩余系,即

$$1,3,7,9,11,13,17,19,21,23,27,29,31,33,37,39$$

共 $\varphi(\varphi(41))=16$ 个数时,6^d 遍历 41 的所有原根。即

$6^1\equiv6 \pmod{41}$　$6^3\equiv11 \pmod{41}$　$6^7\equiv29 \pmod{41}$　$6^9\equiv19 \pmod{41}$

$6^{11}\equiv28 \pmod{41}$　$6^{13}\equiv24 \pmod{41}$　$6^{17}\equiv26 \pmod{41}$　$6^{19}\equiv34 \pmod{41}$

$6^{21}\equiv35 \pmod{41}$　$6^{23}\equiv30 \pmod{41}$　$6^{27}\equiv12 \pmod{41}$　$6^{29}\equiv22 \pmod{41}$

$6^{31}\equiv13 \pmod{41}$　$6^{33}\equiv17 \pmod{41}$　$6^{37}\equiv15 \pmod{41}$　$6^{39}\equiv7 \pmod{41}$

综合定理 5.7 和定理 5.9,构成了求原根的算法基础。

算法 5.1:GetPrimitiveRoot()。

```
/* 原根算法,输出给定素数的所有原根。                          */
/* 输入:素数模 m,以及 m-1的素因子分解 m-1= p1^e1 p2^e2…pk^ek   */
/* 输出:m的所有原根                                           */
{
1.随机选择一个数 a,2≤a≤m-1;
2.对 i从 1到 k执行如下计算:          //求一个原根
    计算 b← a^(m/pi) (mod m);
    如果 b= 1则转到步骤 1;
3.对 d从 1到 m-1,执行如下计算:       //求其他所有原根
    若 gcd(d, m-1)= 1,则输出 a^d (mod m);
}
```

思考 5.4:根据定理 5.9 给出了求阶表的算法。输入为素数模和原根,输出为阶表。

算法 5.2:GetOrder()。

```
/* 输出阶表的算法。        */
/* 输入:素数模 m,原根 a    */
/* 输出:阶表<元素,阶>      */
{
对 i从 1到 m-1执行如下计算:
    Return a^i (mod m), (m-1)/gcd(m-1,i);
}
```

定理 5.10　设 $m>1$ 是整数,a 为整数且 $(a,m)=1$,则

(1) 设 a^{-1} 使得 $a^{-1}a\equiv1 \pmod m$,则 $\mathrm{ord}_m(a^{-1})\equiv\mathrm{ord}_m(a)$。

(2) 若 $b\equiv a \pmod m$,则 $\mathrm{ord}_m(b)=\mathrm{ord}_m(a)$。

证明:(1)因为

$$(a^{-1})^{\mathrm{ord}_m(a)}\equiv(a^{\mathrm{ord}_m(a)})^{-1}\equiv1 \pmod m$$

所以,

$$\mathrm{ord}_m(a^{-1})\mid\mathrm{ord}_m(a)$$

同理可证，$\text{ord}_m(a) \mid \text{ord}_m(a^{-1})$，于是有 $\text{ord}_m(a^{-1})=\text{ord}_m(a)$。

(2) 若 $b \equiv a \pmod m$，则

$$b^{\text{ord}_m(a)} \equiv a^{\text{ord}_m(a)} \equiv 1 \pmod m$$

于是 $\text{ord}_m(b) \mid \text{ord}_m(a)$。同理可证 $\text{ord}_m(a) \mid \text{ord}_m(b)$，于是 $\text{ord}_m(b)=\text{ord}_m(a)$。■

该定理结论(1)也可以从图 5.3 中有直观的解释，3 和 5 互为逆元，5 相当于“反着走”了一圈（如图 5.4 所示）。因为 5“走”1 步到 5，5“走”2 步到 4，以此类推，5“走”6 步到 1。所以 5 的阶为 6。

同理，2 和 4 互为逆元，也可以画个类似的示意图，如图 5.4 所示。

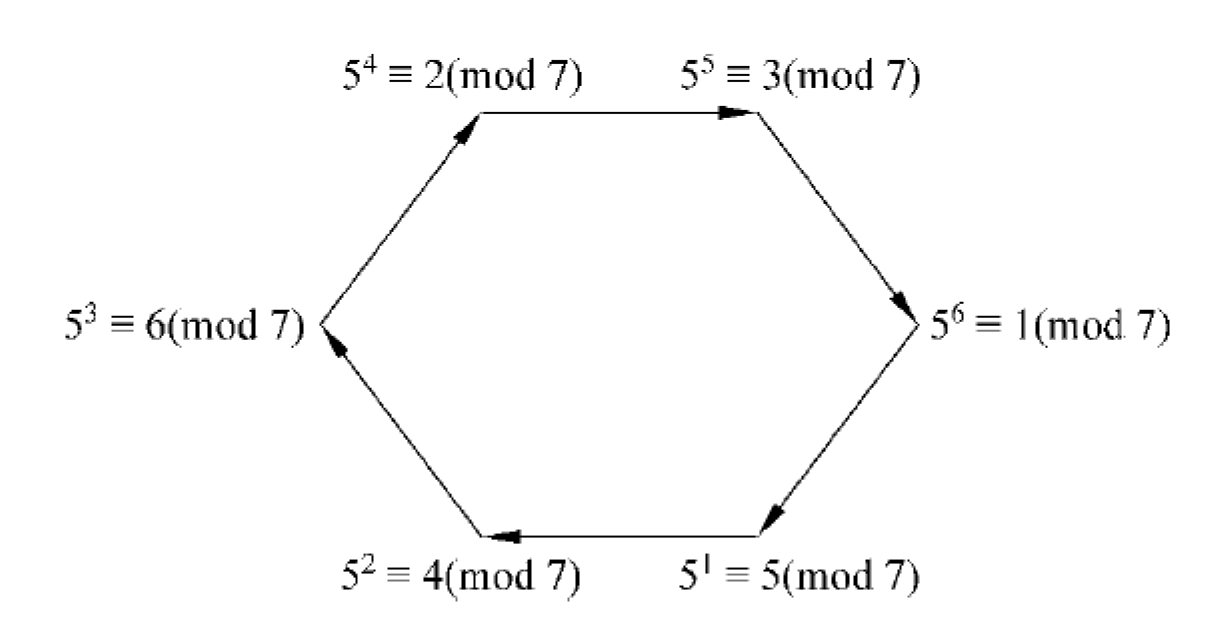


图 5.4　帮助初学者理解的对定理 5.10 的直观解释

下面的定理将原根、阶以及简化剩余系等概念联系到一起。

定理 5.11　设 $m>1$ 是整数，a 为整数且 $(a,m)=1$，则

$$1=a^0, a^1, \cdots, a^{\text{ord}_m(a)-1}$$

模 m 两两不同余。特别地，当 a 是模 m 的原根，即 $\text{ord}_m(a)=\Phi(m)$ 时，这 $\Phi(m)$ 个数组成模 m 的简化剩余系。

证明：反证法。如果 $\text{ord}_m(a)$ 个数中有两个数模 m 同余，则存在整数 $0 \leqslant k, 1 \leqslant \text{ord}_m(a)$，使得

$$a^k \equiv a^1 \pmod m$$

不妨设 $k>1$，则由 $(a,m)=1$ 和定理 2.4，得到

$$a^{k-1} \equiv 1 \pmod m$$

但是 $0 \leqslant k, 1 \leqslant \text{ord}_m(a)$，这与 $\text{ord}_m(a)$ 的最小性矛盾。因此，原结论成立。

当 a 为原根时，即 $\text{ord}_m(a)=\Phi(m)$ 时，共有 $\Phi(m)$ 个数

$$1=a^0, a^1, \cdots, a^{\Phi(m)-1}$$

且模 m 两两不同余，由定理 2.8 知，这 $\Phi(m)$ 个数组成模 m 的简化剩余系。■

在第 6 章将看到，模 m 的简化剩余系构成一个乘法群，其生成元为 a，a 生成了该群中的所有元素。这个群其实还是一个循环群。

5.3　Diffie-Hellman 密钥协商

原根和指数，尤其是循环群的原根，在密码学中有重要的应用。例如，基于离散对数

问题的 Diffie-Hellman 密钥协商协议，以及基于 DH 密钥协商协议的 ElGamal 公钥加密系统。

问题的提出：

在定义 2.7 密码系统的注解中曾经指出对称密码中加密密钥和解密密钥是相同的，因此，对称密码的困难之处有两点：

(1) 密钥的管理。对称加密中加密解密双方使用相同的密钥，因此每一对加密方和解密方就需要一个密钥。而且，密钥必须保密，因此对密钥的存储和管理难度增大。

(2) 当加密方和解密方在不同地理位置时，通信双方如何确定一个秘密的密钥，或者是通信发送方如何将密钥传递给通信接收方，都不是很容易的事情。

思考 5.5：对于共有 n 个通信方的两两通信，需要多少密钥？

对称密码需要的密钥数量：对单个个体而言，需要保存 $n-1$ 个秘密密钥；对总体而言，需要保存 $n(n-1)/2$ 个秘密密钥。

定义 3.2 曾经指出，公钥密码使用公开的公钥进行加密和保密的私钥进行解密。需要的密钥数量：对单个而言，需要保存 n 公钥和 1 个私钥，对总体而言，需要保存 n 个公钥和 n 个私钥。所需密钥的数量减少，而且需要秘密保存的密钥数量大量减少。

因此，需要解决的一个安全问题是：在公开信道上如何协商一个秘密密钥，用于后续的对称密码加密通信。

W. Diffie 与 M. Hellman 利用离散对数问题的困难性，在 1976 年提出了 Diffie-Hellman 密钥协商协议。

首先看离散对数问题中的某种单向性(见图 5.5)：

设 G 是生成元为 g 的 n 阶循环群，则

(1) 给定整数 a，计算元素 g^a 是容易的；

(2) 给定 G 中元素 y，计算整数 x，$1\leqslant x\leqslant n$，使得 $g^x=y$ 通常被认为是困难的(is believed to be hard)。

$g, x \xrightarrow{\text{易}} g^x$

$x \xleftarrow{\text{难}} g^x, g$

图 5.5　离散对数问题的单向性示意图

“被认为是困难的”是指目前公认是困难的，也就是说，目前没有高效率(如多项式时间复杂性)的算法来解决这一问题，但也没有人能够证明其就是困难的。

下面给出一个较严格的定义(需要用到第 6 章中群的概念)。

定义 5.5　乘法群 Z_p^* 上的离散对数问题(Discrete Logarithm Problem，DLP)：给定一个素数 p，乘法群 Z_p^* 上的生成元 g，以及 Z_p^* 上随机选取的元素 y，寻找整数 x，$2\leqslant x\leqslant p-2$，使得 $y=g^x \bmod p$。这个整数 x 记为 $\log_g y$，称为离散对数。

易知，Z_p^* 中元素的个数为 $p-1$，即$\{1,2,\cdots,p-1\}$。

Diffie-Hellman 密钥协商协议是 2 轮协议，即共有 2 个消息在信道中传递。每个通信方发送 1 个消息，并接收 1 个消息。协议的描述如下，如图 5.6 所示。

(1) A 随机选择 $a \in [2, p-2]$，计算 $Y_a = g^a \bmod p$，将 Y_a 发送给 B；

(2) B 随机选择 $b \in [2, p-2]$，计算 $Y_b = g^b \bmod p$，将 Y_b 发送给 A；

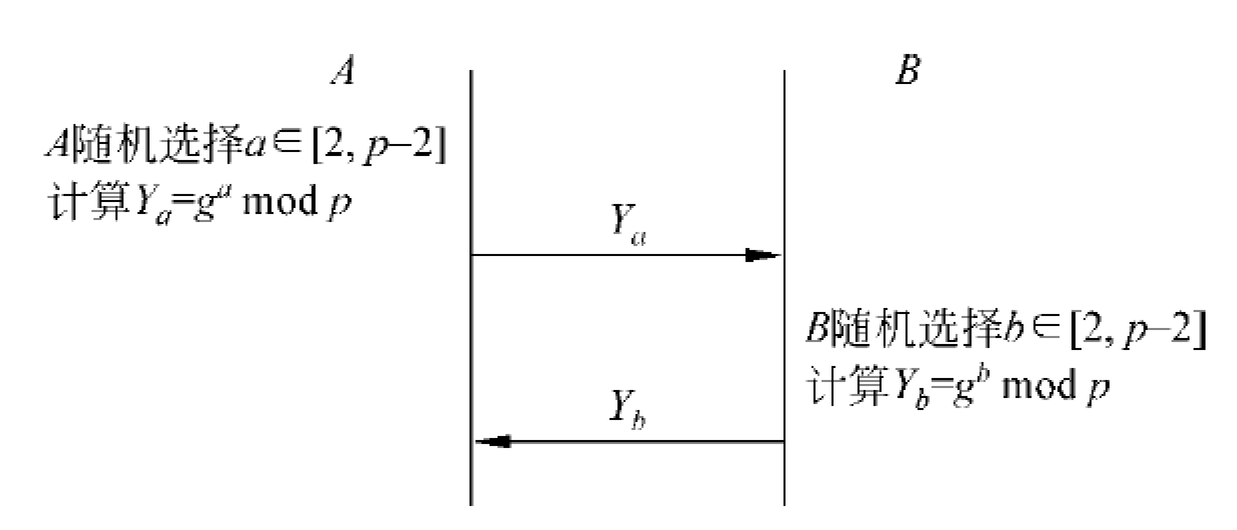


图 5.6　Diffie-Hellman 密钥协商过程

协议交互完成后，A 方以 $Y_b{}^a \bmod p$ 为密钥，B 方以 $Y_a{}^b \bmod p$ 为密钥。A 方和 B 方之间的后续通信可以使用这个密钥进行加密。

例 5.9　$p=41$ 是一个素数，$(Z/41Z)^*$ 是一个乘法循环群，生成元为 6。

实体 A 生成随机整数 $a=6$，计算 $Y_a = 6^6 \bmod 41 = 39$，将 Y_a 发送给 B；

实体 B 生成随机整数 $b=29$，计算 $Y_b = 6^{29} \bmod 41 = 22$，将 Y_b 发送给 A。

A 用自己的 a 和收到的 Y_b 计算：

$$22^6 \pmod{41} = 21$$

B 用自己的 b 和收到的 Y_a 计算：

$$39^{29} \pmod{41} = 21$$

协商后的秘密密钥是 21。

为了对这里例子有更直观的理解，图 5.7 给出一个使用计算机代数系统软件 PARI 作为对这个例子的演示。

```
gp
Reading GPRC: /cygdrive/c/Program Files/PARI/.gprc ...Done.

                 GP/PARI CALCULATOR Version 2.3.4 (released)
          i686 running cygwin (ix86/GMP-4.2.1 kernel) 32-bit version
 compiled: Jul 12 2008, gcc-3.4.4 (cygming special, gdc 0.12, using dmd 0.125)
               (readline v5.2 enabled, extended help available)

                    Copyright (C) 2000-2006 The PARI Group

PARI/GP is free software, covered by the GNU General Public License, and
comes WITHOUT ANY WARRANTY WHATSOEVER.

Type ? for help, \q to quit.
Type ?12 for how to get moral (and possibly technical) support.

parisize = 4000000, primelimit = 500000
(13:52) gp > 6^6 %41
%1 = 39
(13:53) gp > 6^29 %41
%2 = 22
(13:53) gp > 22^6 %41
%3 = 21
(13:53) gp > 39^29 %41
%4 = 21
(13:54) gp >
```

图 5.7　对 Diffie-Hellman 密钥协商(例 5.9)中计算过程的演示

思考 5.6：为什么 A 方和 B 方可以公开地协商出一个秘密的密钥？

A 方计算的 ${Y_b}^a \bmod p$ 等于 ${Y_a}^b \bmod p$。这是因为

$$ {Y_b}^a \bmod p = (g^b \bmod p)^a \bmod p = (g^b)^a \bmod p $$
$$ {Y_a}^b \bmod p = (g^a \bmod p)^b \bmod p = (g^a)^b \bmod p $$

而且协商的过程是公开的，且这种公开的过程是安全的。这是因为：

由于离散对数问题的困难性，敌手即使知道 Y_a，也不能求出 a，即使知道 Y_b，也不能求出 b，没有 a 或者 b，敌人无法计算 ${Y_b}^a \bmod p$ 或者 ${Y_a}^b \bmod p$，即最终 A 和 B 协商出来的秘密密钥。

思考 5.7：为什么是 $a \in [2, p-2]$？

若 $a=1$，则 $Y_a = g^a \bmod p = g$，g 是公开的，于是敌手可以从 Y_a 推出 a；若 $a=p-1$，则 $Y_a = g^a \bmod p = 1$，敌手可以从 Y_a 推出 a。于是，a 为 1 或者 $p-1$ 均不合适。

5.4 ElGamal 公钥密码系统

有了 Diffie-Hellman 密钥协商协议的基础，可以学习 ElGamal 公钥加密系统。

ElGamal 公钥密码系统：

(1) **密钥生成**。

用户 A 随机产生一个大素数 p 以及乘法群 Z_{p^*} 的生成元 g，随机选择 $2 \leqslant a \leqslant p-2$ 作为私钥，计算 $Y_a = g^a \bmod p$。

公钥为 (p, g, Y_a)

私钥为 a

(2) **加密过程**。

用户 B 用 A 的公钥加密消息 m，得到密文。

B 完成如下过程：

随机选择，$2 \leqslant r \leqslant p-2$

计算 $u = g^k \bmod p$，$v = m{Y_a}^k \bmod p$

密文为：(u, v)

(3) **解密过程**。

A 解密密文 (u, v)

得到明文：$m = v/u^a \bmod p$

思考 5.8：ElGamal 加密与 DH 密钥协商协议之间的“神似之处”在哪里？

ElGamal 加密是一种公钥加密，但其实质上是对称加密中的仿射加密加上 DH 密钥协商。图 5.8 给出了一个示意图。

解释如下：

ElGamal 加密后密文有 2 个部分 (u, v)，其中 u 部分为 DH 密钥协商协议部分，v 为

图 5.8 ElGamal 加密的两个部分 DH 密钥协商协议和仿射加密

实质性加密部分。

（1）v 实质性加密部分为仿射加密。表示为

$$v = m \cdot K$$

这里，令 K 为加密方和解密方 DH 密钥协商产生的密钥。即

$$K=(Y_a)^k \bmod p$$

Y_a 为解密方的公钥。

（2）u 为 DH 密钥协商协议部分，即

$$u = g^k \bmod p$$

解密方 A 利用 u 这一部分以及自己的私钥 a，可以生成仿射加密中用到的 K，即

$$K = u^a \bmod p$$

这是因为

$$K = u^a \bmod p = (g^k)^a \bmod p = (g^a)^k \bmod p = (Y_a)^k \bmod p$$

思考 5.9：解密部分的 $v/u^a \bmod p$ 如何计算？

这里，初学者容易误解为“除法”，其实是乘以 u^a 的逆元。有：

$$\begin{aligned} & v/u^a \bmod p \\ &= v \cdot u^{-a} \bmod p \\ &= v \cdot u^{(p-1-a)} \bmod p \end{aligned}$$

例 5.10 设 $p=2579, g=2, d=765$，公钥为 $Y_a=2^{765} \bmod 2579=949$。

加密明文 $m=1299$，秘密选择一个随机整数 $k=853$。计算(u,v)，其中 $u=2^{853} \bmod 2579 = 435$，$v=1299 \cdot 949^{853} \bmod 2579 = 2396$。即密文为(435,2396)。

解密时，明文为 $m=2396/435^{765} \bmod 2579 =1299$。

图 5.9 中的实验给出了计算过程，以加深印象。

其实，可以将离散对数问题从乘法群推广到一般循环群(循环群的定义见 6.2 节)。

定义 5.6 推广的离散对数问题：给定循环群 G，设其阶为 n，生成元为 g，以及 G 上随机选取的元素 y，寻找整数 $x, 1 \leqslant x \leqslant n-1$，使得 $y=g^x$。

由推广的离散对数问题可知，ElGamal 方案可以推广到一般循环群，例如基于椭圆曲线的 ElGamal 方案。

```
gp
Reading GPRC: /cygdrive/c/Program Files/PARI/.gprc ...Done.

                 GP/PARI CALCULATOR Version 2.3.4 (released)
          i686 running cygwin (ix86/GMP-4.2.1 kernel) 32-bit version
 compiled: Jul 12 2008, gcc-3.4.4 (cygming special, gdc 0.12, using dmd 0.125)
               (readline v5.2 enabled, extended help available)

                    Copyright (C) 2000-2006 The PARI Group

PARI/GP is free software, covered by the GNU General Public License, and
comes WITHOUT ANY WARRANTY WHATSOEVER.

Type ? for help, \q to quit.
Type ?12 for how to get moral (and possibly technical) support.

parisize = 4000000, primelimit = 500000
(13:44) gp > 2^853 %2579
%1 = 435
(13:45) gp > 1299*949^853 %2579
%2 = 2396
(13:45) gp > 435^765 %2579
%3 = 2424
(13:45) gp > 1/2424 %2579
%4 = 1980
(13:46) gp > 2396*1980 %2579
%5 = 1299
(13:46) gp >
```

图 5.9　对 ElGamal 公钥密码系统(例 5.10)中计算过程的演示

思　考　题

[1] 计算 2,5,10 模 13 的阶。

[2] 求模 47 的所有原根。

[3] 编写程序实现输出 47 的所有原根。

[4] 编写程序实现输出 1～46 模 47 的阶表。

[5] 求以 6 为底的模 41 的指数表。

[6] 编写程序实现 ElGamal 公钥密码系统。

[7] 利用 PARI 软件验证 DH 密钥协商和 ElGamal 公钥密码系统。

第 6 章 群

本章介绍只有一种运算的代数结构——群。

本章的重点是群的相关概念，循环群。难点是置换群。

6.1 群、子群、同态与同构

定义 6.1 (运算)设 S 是一个非空集合(Set)，那么 $S\times S\rightarrow S$ 的映射叫做 S 上的运算。该运算记为 o(Operator)。

定义 6.2 封闭性(Closure)。对于非空集合 S 以及 S 上的运算 o，S 中任意元素 a 和 b，有 $a\mathrm{o}b\in S$，则运算 o 具有封闭性。

定义 6.3 代数系统(Algebra System)。对于非空集合 S 以及 S 上的运算 o，若运算 o 满足封闭性，则称其为代数系统，记为 $<S,\ \mathrm{o}>$。

例 6.1 $<N,+>$是代数系统，$<N,->$不是代数系统，$<Z,->$是代数系统。

定义 6.4 结合律(Associative)。对于非空集合 S 以及 S 上的运算 o，S 中任意元素 a,b,c，有 $(a\mathrm{o}b)\mathrm{o}c=a\mathrm{o}(b\mathrm{o}c)$，则运算 o 满足结合律。

定义 6.5 满足结合律的代数系统称为半群(Semigroup)。

例 6.2 $<Z,->$不是半群。$<Z,+>$是半群。

定义 6.6 单位元(Identity Element)。设 S 是一个具有运算 o 的非空集合。如果 S 中有一个元素 e 使得

$$ea=ae=a$$

对 S 中所有元素 a 都成立，则称元素 e 为 S 中的单位元。

定义 6.7 独异点(Monoid)。具有单位元的半群称为独异点(也叫做幺半群)。

例 6.3 $<N,+>$是独异点，单位元为 0。$<N,\cdot>$是独异点，单位元为 1。

定义 6.8 逆元(Inverse Element)。设 S 是一个具有运算 o 的有单位元的集合，a 是 S 中的一个元素，如果 S 中存在一个元素 a'，使得

$$aa'=a'a=e$$

则称元素 a 为 S 中的可逆元，称 a' 为 a 的逆元，记为 a^{-1}。

当运算 o 为加法时，这个 a' 叫做 a 的负元，记为 $-a$。

定义 6.9 群(Group)。非空集合 S 中每个元素具有逆元的独异点是群。

例 6.4 m 为合数，$Z_m=Z/mZ=\{0,1,\cdots,m-1\}$。$\cdot$ 表示模 m 的乘法。$<Z_m,\cdot>$

不是群，但<$Z_m{}^*$，·>是群，这里 $Z_m{}^*$ 表示模 m 的最小非负简化剩余系。

定义 6.10　交换律(Commutative Law)。设 S 是一个具有运算 o 的非空集合，如果对 S 中任意元素 a 和 b，有

$$a\text{o}b = b\text{o}a$$

则称运算 o 具有交换律。

定义 6.11　群中运算具有交换律的群叫做交换群，或者 Abel 群。

例 6.5　<Z，+>是交换群。有理数集合 Q，实数集 R，复数集 C 对于通常意义下的加法+和乘法·而言，<$Q/\{0\}$，·>，<$R/\{0\}$，·>，<$C/\{0\}$，·>是交换群。

图 6.1 以一种"递进"的方式给出概念间的联系，非正式地说，显示了群定义中类似于"进化完善"的过程，便于理解和记忆。

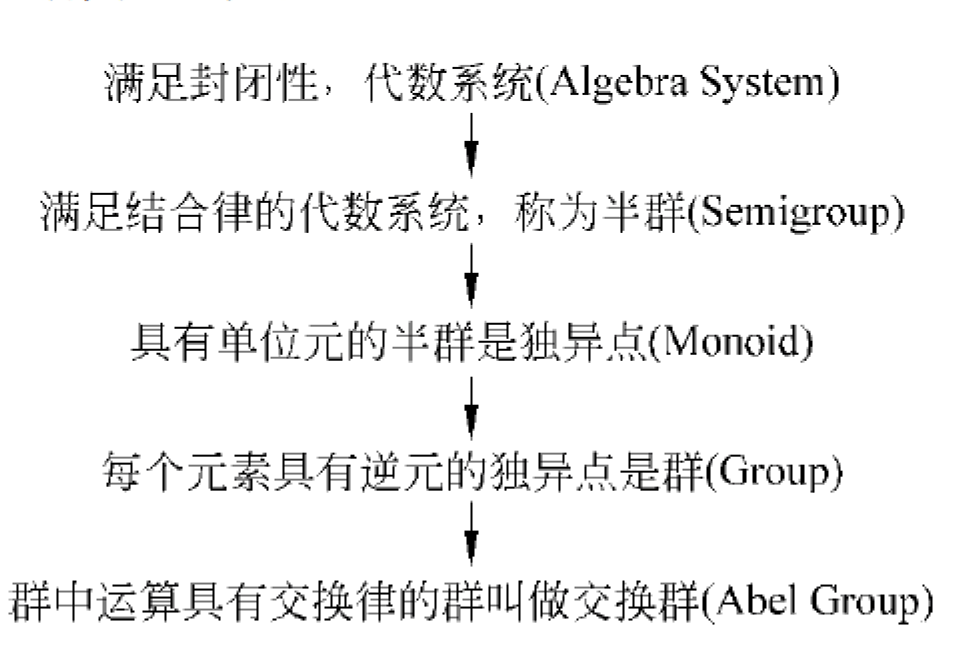


图 6.1　用"递进"的方式给出概念间的联系便于理解群的"进化完善"过程

定义 6.12　若群 G 中含有有限个元素，则称群 G 为有限群；若群 G 中含有无限多个元素，则称群 G 为无限群。一个有限群 G 中的元素个数称为群的阶，记为 $|G|$。

例 6.6　设 n 是一个正整数，集合为 $Z/nZ=\{0,1,2,\cdots,n-1\}$，集合 Z/nZ 上的运算为加法

$$a\oplus b=(a+b)\ (\text{mod}\ n)$$

其中 $a\ (\text{mod}\ n)$是整数 a 模 n 的最小非负剩余。G=<Z/nZ，$\oplus$>构成一个交换加群。$|G|=n$。

例 6.7　设 n 是一个合数，集合$(Z/nZ)^*=\{a|a\in nZ,(a,n)=1\}$，集合$(Z/nZ)^*$上的运算为乘法

$$a\odot b=(a+b)\ (\text{mod}\ n)$$

G=<$(Z/nZ)^*$，$\odot$>构成一个交换乘法群。$|G|=\Phi(n)$。

下面引入幂次的概念。

设 n 是正整数，如果 $a_1=a_2=\cdots=a_n=a$，则记 $a_1a_2\cdots a_n=a^n$，称之为 a 的 n 次幂，特别地，定义 $a^0=e$ 为单位元，$a^{-n}=a^{-1}$的 n 次幂。

由群的结合律可以得到，a 是群 G 中的任意元，则对任意的整数 m,n，有

$$a^ma^n=a^{m+n},(a^m)^n=a^{mn}$$

下面引入元素的阶(order)的概念。

定义 6.13　设 $a\in$ 群 G，e 是 G 的单位元，若任意 $k\in N$，$a^k\neq e$，则称 a 的阶为无穷大，记作 $|a|=\infty$，若存在 $k\in N$ 使得 $a^k=e$，则称 $\min\{k|k\in N,a^k=e\}$为 a 的阶。若 a 的阶是

n，则记作$|a|=n$。

下面讨论两个群之间的关系(由群中集合的包含关系确定)。

定义 6.14 设 H 是群 G 的一个非空子集合，如果对于群 G 的运算，H 成为一个群，则 H 叫做群 G 的子群(Subgroup)，记作 $H \leqslant G$。

$H=\{e\}$ 和 $H=G$ 都是群 G 的子群，叫做群 G 的平凡子群。

G 的子群 H 叫做群 G 的真子群，如果 H 不是群 G 的平凡子群。

例 6.8 $3Z=\{\cdots,-6,-3,0,3,6,\cdots\}$ 是 Z 的子群。$6Z=\{\cdots,-12,-6,0,6,12,\cdots\}$ 是 $3Z$ 的子群，即 $6Z \leqslant 3Z$。

例 6.9 设 n 是一个正整数，则 $nZ=\{nk \mid k \in Z\}$ 是 Z 的子群。

除了用定义来判定外，子群的判定还可以根据如下更为简单的判定定理。

定理 6.1 设 H 是群 G 的一个非空子集合，则 H 是群 G 的子群的充要条件是对任意的 a,b 是对任意的 $a,b \in H$，有 $ab^{-1} \in H$。

证明：必然性是显然的，下面证明充分性。

因为 H 非空，所以 H 中有元素 a，根据假设，有 $e=aa^{-1} \in H$，因此 H 中有单位元。对于 $e \in H$ 及任意 a，再应用假设，有 $a^{-1}=ae \in H$，即 H 中每个元素 a 在 H 中均有逆元。对于任意 $a,b \in H$，由 $ab=a(b^{-1})^{-1} \in H$，可知 H 对乘法运算封闭，因此 H 是群 G 的子群。 ■

下面讨论群与群之间的两种特殊的映射。

定义 6.15 设 G 和 G' 是两个群，f 是 G 到 G' 的一个映射，如果对任意的 $a,b \in G$，都有

$$f(ab)=f(a)f(b) \tag{6.1}$$

那么，f 叫做 G 到 G' 的一个同态(homomorphism)。

如果 f 是一对一的(Injective)，则称 f 为单同态；如果 f 是满的(Surjective)，则称 f 为满同态；如果 f 是一一对应的(Bijective)，则称 f 为同构(Isomorphism)。

当 $G=G'$ 时，同态 f 叫做自同态，同构 f 叫做自同构。

思考 6.1 同态映射中的运算是否有区别。

需要注意的是，(6.1)式中左边 ab 之间的运算是 G 群中的运算，右边 $f(a)f(b)$ 之间的运算是 G' 群中的运算。

定义 6.16 设 G、G' 是两个群，如果存在一个 G 到 G' 的同构，称 G 到 G' 同构，记作 $G \cong G'$。

例 6.10 加群 Z 到乘群 $R^*=R\backslash\{0\}$ 的映射 $f: a \to e^a$ 是 Z 到 R^* 的一个同态。

例 6.11 加群 Z 到加群 $Z_n=Z/nZ$ 的映射 $f: n \to k(k \in \{0,1,\cdots,n-1\})$ 是 Z 到 Z_n 的一个同态。

6.2 循环群

本节讨论循环群，其中很多定理是第 5 章中定理的一般化。

定理 6.2 设 G 是一个群，$\{Hi\}_{i \in I}$ 是 G 的一族子群，则 $\bigcap_{i \in I} Hi$ 是 G 的一个子群。

例 6.12　$2Z$ 是 Z 的子群，$3Z$ 是 Z 的子群。$2Z\cap 3Z=6Z$ 是 Z 的子群。

定理 6.3　设 G 是一个群，X 是 G 的非空子集，$\{Hi\}_{i\in I}$ 是 G 的包含 X 的所有子群，则 $\bigcap_{i\in I}Hi$ 是 G 的由 X 生成的子群，记为 $<X>$。

例 6.13　$X=6Z$，G 中包含 X 的所有子群为 $2Z,3Z,6Z$。$2Z\cap 3Z\cap 6Z=6Z$，$6Z$ 生成的子群记为 $<6Z>$。

实际上，$<X>$ 是 G 中包含 X 的最小子群。

X 的元素称为子群 $<X>$ 的生成元。如果 $X=\{a_1,\cdots,a_n\}$，则记 $<X>$ 为 $<a_1,a_2,\cdots,a_n>$。如果 $G=<a_1,a_2,\cdots,a_n>$，则称 G 为有限生成的。特别地，如果 $G=<a>$，则称 G 为由 a 生成的循环群。

定理 6.4　设 G 是一个群，X 是 G 的非空子集，则由 X 生成的子群为

$$<X>=\{a_1^{n_1}\cdots a_t^{n_t}\mid t\in N,a_i\in G,n_i\in Z,1\leqslant i\leqslant t\}$$

特别地，对任意的 $a\in G$，有

$$<a>=\{a^n\mid n\in Z\}$$

例 6.14　设 $G=<g>=\{g^r\mid g^r\neq 1,1\leqslant r<n,g^n=1\}$，$G$ 是 n 阶循环群，则

$$<g^d>=\{g^{dk}\mid k\in Z\}$$

这一定义和第 5 章中原根和阶的定义形成了某种“呼应”。

定理 6.5　加群 Z 的每个子群 H 都是循环群，且有 $H=<0>$ 或者 $H=<m>=mZ$，其中 m 是 H 中的最小正整数。如果 $H\neq<0>$，则 H 是无限的。

定理 6.6　每个无限循环群同构于加群 Z，每个阶为 m 的有限循环群同构于加群 Z/mZ。

此定理说明，循环群的构造完全取决于它的生成元，当生成元的阶是无限大时，它们互相同构且都与 $<Z,+>$ 同构。当生成元的阶是 m 时，它们互相同构且都与 $<Z/mZ,\oplus_m>$ 同构（这里 $\oplus_m$ 表示模 m 加法）。因此，从同构的观点看，循环群只有两种：整数加群和模 m 的剩余类加群。于是，对循环群的研究可转移到对这两个群的研究上来。

接下来由循环子群的阶给出群元素阶的等价定义。

定义 6.17　设 G 是一个群，$a\in G$，则子群 $<a>$ 的阶称为元素 a 的阶，记 $|a|$。

定理 6.7　设 G 是一个群，$a\in G$。如果 a 是无限阶，则：

(1) $a^k=e$ 当且仅当 $k=0$；

(2) 元素 $a^k(k\in Z)$ 两两不同；

如果 a 是有限阶 $m>0$，则：

(3) m 是使得 $a^m=e$ 的最小正整数；

(4) $a^k=e$ 当且仅当 $m\mid k$；

(5) $a^r=a^k$ 当且仅当 $r\equiv k\ (\mathrm{mod}\ m)$；

(6) 元素 $a^k(k\in Z/mZ)$ 两两不同；

(7) $<a>=\{a,a^2,\cdots,a^{m-1},a^m=e\}$；

(8) 对任意整数 $1\leqslant d\leqslant m$，有 $|a^d|=\dfrac{m}{(m,d)}$

定理 6.8　循环群的子群是循环群。

定理 6.9 设 G 是循环群，$G=\langle a\rangle$，如果 G 是无限的，则 G 的生成元为 a 和 a^{-1}，如果 G 是有限阶 m，则 a^k 是 G 的生成元当且仅当 $(k,m)=1$。

思考 6.2：这个定理是否"似曾相识"呢。

这个可以视为前面学到的定理 5.10 的一般化。

6.3 置换群

6.3.1 置换群的概念

设 S 是一个非空集合，G 是 S 到自身的所有一一对应的映射组成的集合，则对于映射的复合运算，G 构成一个群，叫做对称群。

对称群的单位元是恒等映射。G 中的元素叫做 S 的一个置换。

当 S 是 n 元有限集时，G 叫做 n 元对称群，记作 Sn。

设 $S=\{1,2,\cdots,n\}$ 及其对称群 Sn 中的元素 σ，有 $\sigma:i\to\sigma(i)$。通常将置换 σ 写成

$$\sigma=\begin{pmatrix}1 & 2 & \cdots & n-1 & n\\ \sigma(1) & \sigma(2) & \cdots & \sigma(n-1) & \sigma(n)\end{pmatrix}=\begin{pmatrix}i_1 & i_2 & \cdots & i_{n-1} & i_n\\ \sigma(i_1) & \sigma(i_2) & \cdots & \sigma(i_{n-1}) & \sigma(i_n)\end{pmatrix}$$

例如，

$$\sigma=\begin{pmatrix}1 & 2 & 3 & 4 & 5 & 6\\ 5 & 4 & 3 & 6 & 1 & 2\end{pmatrix}=\begin{pmatrix}1 & 5 & 2 & 4 & 6 & 3\\ 5 & 1 & 4 & 6 & 2 & 3\end{pmatrix}$$

图 6.2 给出了 σ 映射的图示。

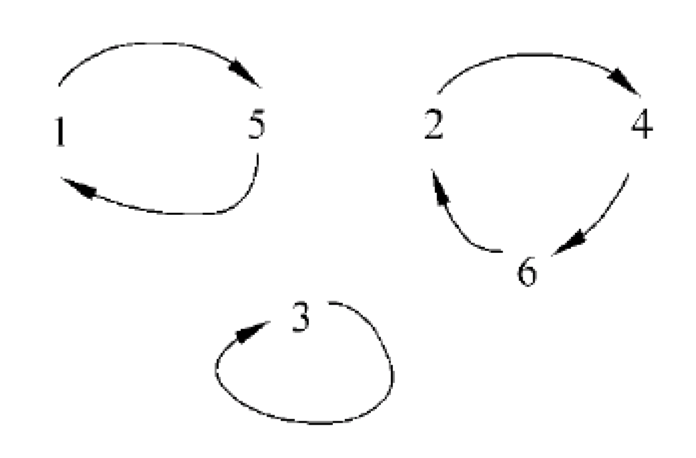


图 6.2 σ 映射的图示

例 6.15 $\sigma=\begin{pmatrix}1 & 2 & 3 & 4 & 5 & 6\\ 6 & 5 & 4 & 3 & 1 & 2\end{pmatrix}$，$\tau=\begin{pmatrix}1 & 2 & 3 & 4 & 5 & 6\\ 5 & 6 & 4 & 2 & 3 & 1\end{pmatrix}$，求 $\sigma\tau,\tau\sigma,\sigma^{-1}$。

解答：
$$\sigma\tau=\begin{pmatrix}1 & 2 & 3 & 4 & 5 & 6\\ 6 & 5 & 4 & 3 & 1 & 2\end{pmatrix}\begin{pmatrix}1 & 2 & 3 & 4 & 5 & 6\\ 5 & 6 & 4 & 2 & 3 & 1\end{pmatrix}$$
$$=\begin{pmatrix}5 & 6 & 4 & 2 & 3 & 1\\ 1 & 2 & 3 & 5 & 4 & 6\end{pmatrix}\begin{pmatrix}1 & 2 & 3 & 4 & 5 & 6\\ 5 & 6 & 4 & 2 & 3 & 1\end{pmatrix}$$
$$=\begin{pmatrix}1 & 2 & 3 & 4 & 5 & 6\\ 1 & 2 & 3 & 5 & 4 & 6\end{pmatrix}$$
$$\tau\sigma=\begin{pmatrix}1 & 2 & 3 & 4 & 5 & 6\\ 5 & 6 & 4 & 2 & 3 & 1\end{pmatrix}\begin{pmatrix}1 & 2 & 3 & 4 & 5 & 6\\ 6 & 5 & 4 & 3 & 1 & 2\end{pmatrix}$$

$$=\begin{pmatrix}6&5&4&3&1&2\\1&3&2&4&5&6\end{pmatrix}\begin{pmatrix}1&2&3&4&5&6\\6&5&4&3&1&2\end{pmatrix}$$

$$=\begin{pmatrix}1&2&3&4&5&6\\1&3&2&4&5&6\end{pmatrix}$$

$$\sigma^{-1}=\begin{pmatrix}6&5&4&3&1&2\\1&2&3&4&5&6\end{pmatrix}=\begin{pmatrix}1&2&3&4&5&6\\5&6&4&3&2&1\end{pmatrix}$$

定理 6.10　n 元置换全体组成的集合 Sn 对置换的乘法构成一个群，其阶为 $n!$。

定理 6.11　任意一个置换都可以表示成一些不相交轮换的乘积，在不考虑乘积次序的情况下，该表达式是唯一的。

例 6.16　$\sigma=\begin{pmatrix}1&2&3&4&5&6\\6&5&2&1&3&4\end{pmatrix}=(1\quad 6\quad 4)(2\quad 5\quad 3)$

$=(1\quad 4)(1\quad 6)(2\quad 3)(2\quad 5)$

由 n 元置换构成的群叫做 n 元置换群。

例 6.17　设 $\sigma=(1,2,3)$，则循环群 $G=\langle\sigma\rangle=\{e,(1,2,3),(1,3,2)\}$ 是 3 元置换群。

定理 6.12　（凯莱定理）设 G 是一个 n 元群，则 G 同构于一个 n 元置换群。

该定理是群论中一个重要定理，它揭示了置换群与抽象群之间的关系。

6.3.2　置换群的应用*

群的应用在 5.3 节和 5.4 节中已经给出了一个常见的重要应用。第 9 章将给出循环群在椭圆曲线上的构造以及应用。本节主要讨论置换群在转轮密码机中的应用。

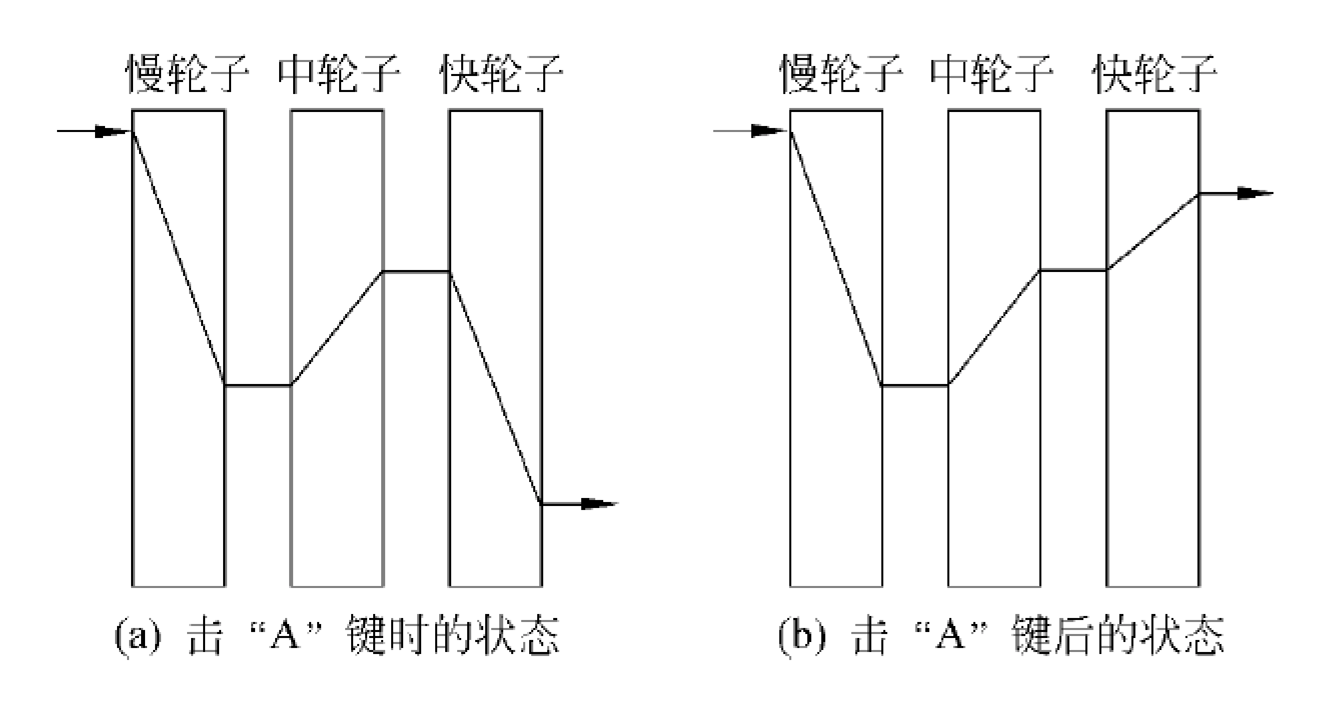
古典密码体制实际上可以分为人工加密和机械加密两种。从 19 世纪 20 年代开始，人们逐渐发明各种机械加解密设备用来处理数据的加解密运算，最典型的设备是转轮密码机（Rotor Machine）。转轮密码机由一组布线轮和转轮轴组成的灵巧复杂的机械装置，可以实现长周期的多表代换，且加密和解密的过程由机械自动快速完成。

1918 年发明德国发明家 Artrhur Scherbius 发明了名叫 ENIGMA 的转轮密码机，意为“谜”，后来被德国装备军队使用，又大大加强了基本设计[①]。转轮密码机的使用极大提高了加解密速度，同时抗攻击性能有很大的提高，在第二次世界大战中有着广泛的应用，是密码学发展史上的一个里程碑。

转轮密码机由一个输入的键盘和一组转轮组成，每个转轮上有 26 个字母的输入引脚，以及 26 个字母的输出引脚，输入输出关系由内部连线决定。以简化版的转轮密码机为例，假设有 3 个转轮，从左到右分别为慢轮子、中轮子、快轮子（例如通过齿轮控制）。按下某一键时，键盘输入的明文电信号从慢轮子进入转轮密码机，轮子之间传递电信号，最后从快轮子输出密文。每次击键后，快轮子就转动一格，这样就改变了中轮子和快轮子之间的对应关系。两次连续按“A”键，得到的密文结果不一样，于是形成多表代换关系。图 6.3 给出其原理的示意图，图 6.4 给出一个实例。在首次击“A”键时，输出“E”，输出后

① 另一个著名的转轮密码机时美军的 Haglin 密码机，是瑞典 Haglin 发明的，在二战时被盟军广泛使用。同时，二战时日军的“紫密”和“兰密”也是转轮密码机。

快轮子转动一格，导致中轮子和快轮子的接触点变化，再通过内部连线，改变了输出。再次击“A”键，输出为“B”(快轮子与 24 对应的是 18，18 对应“B”)。



(a) 击“A”键时的状态　　(b) 击“A”键后的状态

图 6.3　转轮密码机原理

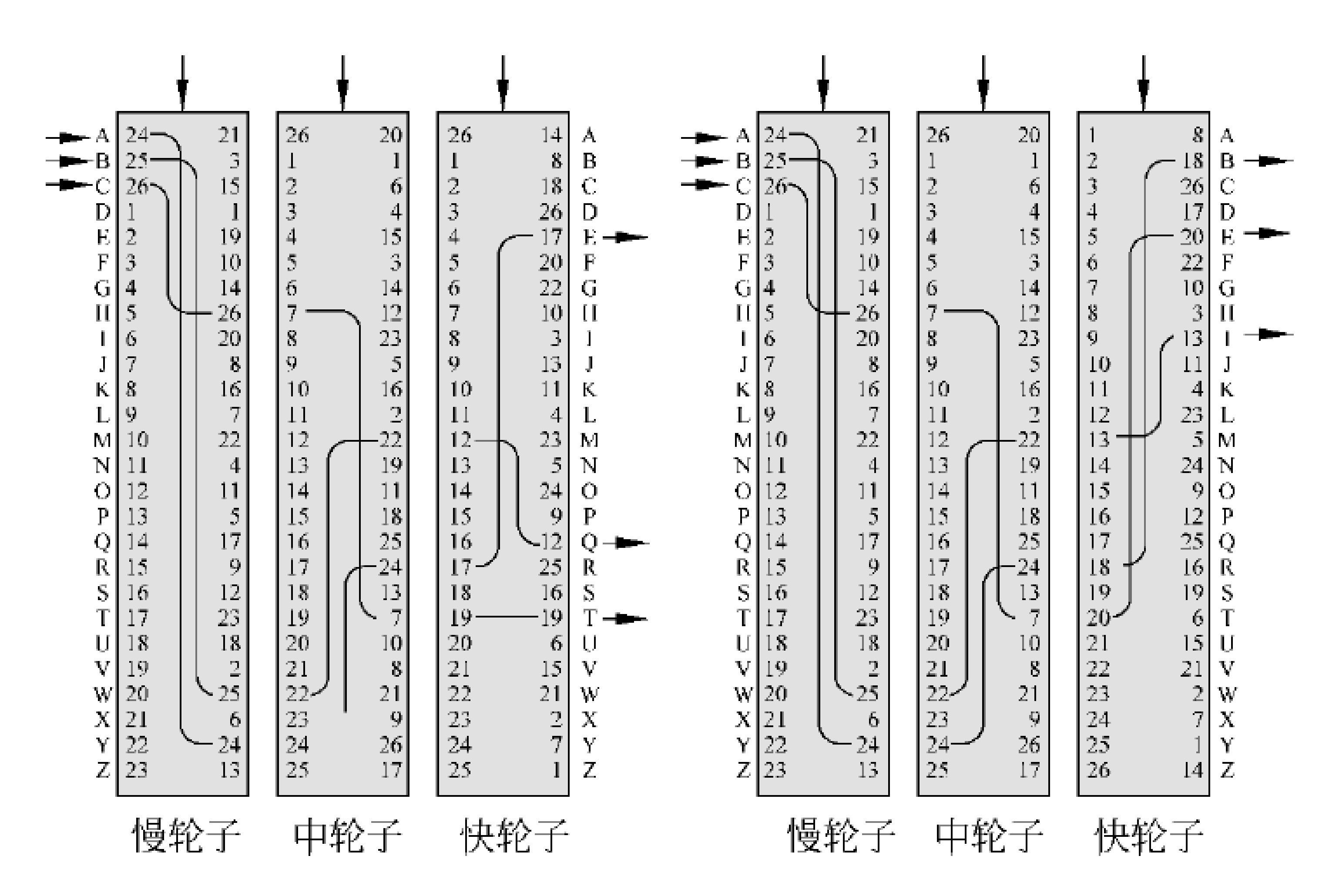


图 6.4　转轮密码机的一个实例

下面解释其原理。每个转轮相当于一个置换。多个转轮如果转速相同，则相当于一个转轮，还是一个置换的效果。快轮子转动一圈(即 26 格)，中轮子转动一格；中轮子转动一圈(即 26 格)，慢轮子转动一格。由于多个转轮的转速不一样，转轮之间的对应关系在每次按键后均改变，形成了所谓的“多表代换”。二战期间阿兰·图灵(Alan Turning)参与了英国政府的破译行动，成功利用 Enigma 使用上的缺陷破译了密码，一定程度上改变了战争的局势。

思考 6.3：轮子的转动是在改变代换方式吗?

如果把轮子的转动导致的轮子间映射关系的改变视为代换(Substitution)，把轮子内部的连线视为置换(Permutation)，则转轮机可视为后来在 DES 分组密码中用到的代换置换网络(SP 网络)的雏形，(可能是信息论之父香农 Shannon 后来提出设计密码的混淆思想和扩散思想的源泉)。转轮机中的多个转轮提高了安全性(增大了密钥空间)，(也可能是后来 Shannon 提出乘积密码的源泉)。

分组密码(如 DES 和 AES)的设计中，都大量使用了置换操作。

思　考　题

[1] 证明：群 G 是交换群的充要条件是对任意 $a, b \in G$，有 $(ab)^2 = a^2 b^2$。

[2] $p=7$，构造乘群 $F_p^* = F_p \setminus \{0\}$ 的乘法表和加群 $Z/(p-1)Z$ 的加法表。

[3] 设 p 是奇素数，证明乘群 $F_p^* = F_p \setminus \{0\}$ 是同构于加群 $Z/(p-1)Z$ 的循环群。

[4] $\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 3 & 4 & 5 & 6 & 1 \end{pmatrix}$，$\tau = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 5 & 3 & 4 & 2 & 6 & 1 \end{pmatrix}$，求 $\sigma\tau$，$\tau\sigma$，σ^{-1}。

第7章 环与域

上一章介绍的群是只有一种运算的代数系统，本章考虑具有两种运算的代数系统。本章的重点是环和域的概念，难点是多项式环和多项式域。

7.1 环

7.1.1 环和域的概念

定义 7.1 设 $R=<S, o1,o2>$，S 是具有两种运算(o1 和 o2)的非空集合，如果：

(1) $<S, o1>$构成一个交换群；

(2) $<S, o2>$构成半群；

(3) o1，o2 满足分配律(Distributive Law)，即：对任意 $a,b,c\in S$，有

$$(a \text{ o1 } b)\text{o2 } c = a \text{ o2 } c \text{ o1 } b \text{ o2 } c \text{（右分配律）}$$

$$a \text{ o2}(b \text{ o1 } c) = a \text{ o2 } b \text{ o1 } a \text{ o2 } c \text{（左分配律）}$$

则 R 称为环(Ring)。

(4) 若环 R 中$<S, o2>$构成交换半群。则 R 称为交换环。

(5) 若环 R 中$<S, o2>$构成独异点，则 R 称为有单位元的环(也称为含幺环)。

例 7.1 $<Z,+,\cdot>$是一个有单位元的交换环。$<Z,+>$是交换加群，零元为 0，a 的负元为 $-a$。$+$，$\cdot$ 满足分配律。$<Z,\cdot>$是半群。因此，$<Z,+,\cdot>$是一个环。而且，$<Z,\cdot>$具有单位元 1，且 $\cdot$ 满足交换律。因此，是有单位元的交换环。

定义 7.2 设 $R=<S,+,\cdot>$是一个环，如果存在 $a,b\in S$，满足 $a\neq 0,b\neq 0$，但 $a\cdot b=0$，则称环 R 为有零因子环，称 a 为 R 的左零因子，b 为 R 的右零因子，否则称 R 为无零因子环。

例 7.2 整数环$<Z,+,\cdot>$、有理数环$<Q,+,\cdot>$、实数环$<R,+,\cdot>$、复数环$<C,+,\cdot>$均为无零因子环。

例 7.3 对于合数 n，Z_n 为有零因子环。

例 7.4 对于素数 p，Z_p 为无零因子环。

例 7.5 $<Z/6Z=\{0,1,2,3,4,5\},\cdot\ (\text{mod } 6)>$是一个有零因子环，因为 $2\cdot 3=0$。

在无零因子环中，乘法消去律成立。即非空集合中任意 $a,b,c,a\neq 0$，有

$$a\cdot b=a\cdot c \Rightarrow b=c$$

$$b\cdot a=c\cdot a \Rightarrow b=c$$

定义 7.3　有单位元的无零因子的交换环叫做整环(Integral Domain)。

例 7.6　整数环 $<Z,+,\cdot>$、有理数环 $<Q,+,\cdot>$、实数环 $<R,+,\cdot>$、复数环 $<C,+,\cdot>$ 都是整环,而 $2Z$(偶数环)没有单位元故不是整环,Z_n(n 为合数)有零因子故不是整环。

定义 7.4　一个环 R 叫做除环,当环 R 有以下特征时:

(1) R 中至少包含一个非零元(即 R 中至少有两个元素);

(2) R 有单位元;

(3) R 的每一个非零元有逆元。

应当注意到,除环的概念中,没有要求满足乘法交换律。

其实除环就是指环中非零元在乘法运算下构成群。

定义 7.5　交换除环叫做域(Field)。

例 7.7　有理数集 Q、实数集 R、复数集 C 关于数的加法和乘法都构成域。但是整数环只能构成整环,不能构成域。

定义 7.6　如果交换环 R 对于加法构成一个交换群,$R^{*}=R\backslash\{0\}$ 对于乘法构成一个交换群,则交换环 R 为一个域。

例 7.8　设 p 是素数,则 $<Z_p, +\ (\mathrm{mod}\ p), \cdot\ (\mathrm{mod}\ p)>$ 是域。

注意,域一定是整环,但整环不一定是域,但有限整环一定是域。

定义 7.7　只包含有限个元素的环(域)称为有限环(域),其元素的个数称为该环(域)的阶。有限域又叫伽罗瓦域(Galois Field)。

为了便于理解和记忆,表 7.1 给出了上述概念之间的递进关系。非正式地,图 7.1 给出了环概念的“进化完善”过程。

表 7.1　环概念之间的“递进”关系

<S,o1,o2>	<S,o1>	<S,o2>	o1,o2
环	交换群	半群	分配律
交换环	交换群	交换半群	分配律
有单位元的环	交换群	独异点	分配律
有单位元的交换环	交换群	交换独异点	分配律
无零因子环	交换群	无零因子半群	分配律
整环	交换群	无零因子交换独异点	分配律
除环	交换群	有非零元且非零元有逆元的独异点,或者 $<S\backslash\{0\},o2>$ 为群	分配律
域	交换群	$<S\backslash\{0\},o2>$ 为交换群	分配律

例 7.9　一个典型的域是 $<F_2,+,\cdot>$,$<F_2,+>$ 为一个交换群。$<F_2\backslash\{0\},\cdot>$ 为一个交换群。图 7.2 给出了其运算表。

同态概念可以扩展到环。

定义 7.8　设 R,R' 是两个环,称映射 $f: R\to R'$ 为环同态,如果 f 满足下列条件:

环<S, o1, o2> = 交换群<S, o1> + 半群<S, o2> + 分配律o1, o2
交换环<S, o1, o2> = 交换群<S, o1> + 交换半群<S, o2> + 分配律o1, o2
有单位元的环<S, o1, o2> = 交换群<S, o1> + 独异点<S, o2> + 分配律o1, o2
有单位元的交换环<S, o1, o2> = 交换群<S, o1> + 交换独异点<S, o2> + 分配律o1, o2
无零因子环<S, o1, o2> = 交换群<S, o1> + 无零因子半群<S, o2> + 分配律o1, o2
整环<S, o1, o2> = 交换群<S, o1> + 无零因子交换独异点<S, o2> + 分配律o1, o2
除环<S, o1, o2> = 交换群<S, o1> + <S\{0}, o2>为群+ 分配律o1, o2
域<S, o1, o2> = 交换群<S, o1> + <S\{0}, o2>为交换群+ 分配律o1, o2=交换除环=有限整环

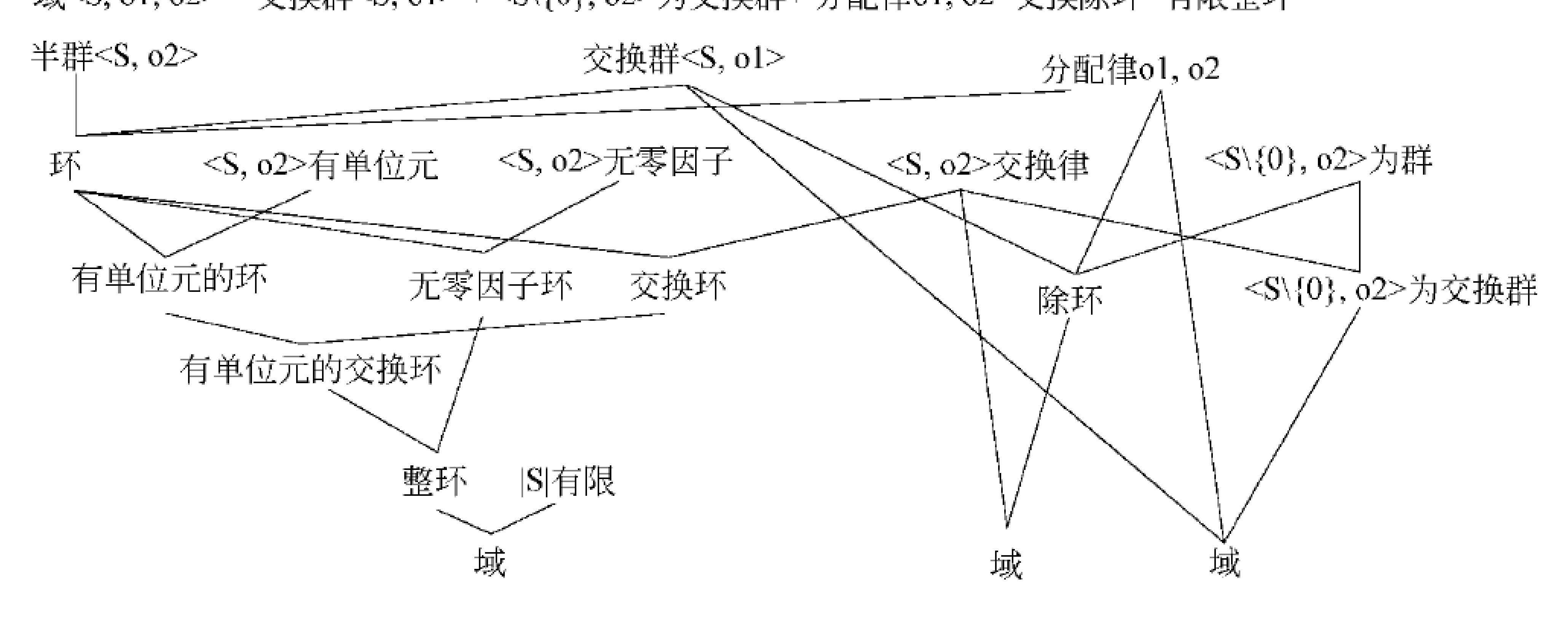


图 7.1　环的“进化完善”过程示意图

(1) 对任意的 $a,b\in R$，都有 $f(a+b)=f(a)+f(b)$

(2) 对任意的 $a,b\in R$，都有 $f(ab)=f(a)f(b)$

如果 f 是一对一的，则称 f 为单同态；如果 f 是满的，则称 f 是满同态；如果 f 是一一对应的，则称 f 为同构。

+	0	1
0	0	1
1	1	0

•	0	1
0	0	0
1	0	1

图 7.2　域 $<F_2,+,\cdot>$ 的运算表

定义 7.9　设 R,R' 是两个环，称 R 与 R' 同构，如果存在一个 R 到 R' 的同构。

定义 7.10　设 R 是一个环，如果存在一个最小正整数 n 使得对任意 $a\in R$，都有 $na=0$，则称环 R 的特征为 n，记为 $\mathrm{char}(R)=n$；如果不存在这样的正整数，则称环 R 的特征为零，记为 $\mathrm{char}(R)=0$。

定理 7.1　设 R 为一整环，则 $\mathrm{char}(R)=0$ 或者 $\mathrm{char}(R)=p$，其中 p 为一素数。

推论 1：整环 R 的加法群中每一非零元的阶或都为无穷，或都为一素数。

推论 2：整环 R 的特征即为单位元“1”在 R 的加法群中的阶。

定理 7.2　如果域 K 的特征不为零，则其特征必为素数。

证明：设域 K 的特征为 n，如果 n 不是素数，则存在整数 $1<n_1,n_2<n$，使得 $n=n_1n_2$，从而 $(n_1 1_k)(n_2 1_k)=(n_1n_2)1=0$，因为域 K 无零因子，所以 $n_1 1_k=0$ 或者 $n_2 1_k=0$，这与特征 n 的最小性矛盾。 ■

定理 7.3　设 R 是有单位元的交换环，如果环 R 的特征是素数 p，则对任意 $a,b\in R$，有

$$(a+b)^p=a^p+b^p$$

证明留作练习。

与子群类似，可以给出子环，子域的概念。

定理 7.4　设 R' 是环 R 的非空子集，如果对于环 R 的运算，R' 也构成一个环，则 R' 叫做 R 的子环。

定理 7.5　设 K' 是环 K 的非空子集，如果对于域 K 的运算，K' 也构成一个环，则 K' 叫做 K 的子域。

定义 7.11　一个域叫做素域，如果它不含真子域。

例 7.10　有理数域 Q 是素域。$F_p=Z/pZ$ 是素域，其中 p 为素数。

定理 7.6　设 F 是一个域，如果 F 的特征为 0，则 F 有一个与 Q 同构的素域。如果 F 的特征为 p，则 F 有一个与 F_p 同构的素域。

7.1.2　多项式环

设

$$f(x)=a_nx^n+\cdots+a_1x+a_0,\ g(x)=b_nx^n+\cdots+b_1x+b_0\in R[X]$$

在 $R[X]$ 上定义加法：

$$(f+g)(x)=(a_n+b_n)x^n+\cdots+(a_1+b_1)x+(a_0+b_0)$$

则 $R[X]$ 对于该加法构成一个交换加群。

零元为 0，$f(x)$ 的负元为

$$-f(x)=(-a_n)x^n+\cdots+(-a_1)x+(-a_0)$$

设

$$f(x)=a_nx^n+\cdots+a_1x+a_0,\ a_n\neq 0$$
$$g(x)=b_nx^n+\cdots+b_1x+b_0,\ b_n\neq 0$$

在 $R[X]$ 上定义乘法：

$$(f\cdot g)(x)=c_{n+m}x^{n+m}+\cdots+c_1x+c_0$$

其中，$c_k=\sum\limits_{i+j=k}a_ib_j=a_kb_0+a_{k-1}b_1+\cdots+a_1b_{k-1}+a_0b_k\ (0\leqslant k\leqslant n+m)$，即

$$c_{n+m}=a_nb_m,c_{n+m-1}=a_nb_{m-1}+a_{n-1}b_m,\cdots,c_0=a_0b_0$$

则 $R[X]$ 中的单位元为 1。

$R[X]$ 对于上述加法运算和乘法运算构成一个整环。

例 7.11　系数为 F_2 上的多项式环记为 $F_2[X]$，其中多项式 $f(x)=x^2+x+1$，$g(x)=x+1$，求 $g(x)^2$，$f(x)g(x)$。

解答：$g(x)^2=(x+1)^2=x^2+1$，$f(x)g(x)=(x^2+x+1)(x+1)=x^3+x^2+1$。

设 $f(x)=a_nx^n+\cdots+a_1x+a_0$，$a_n\neq 0$，则称多项式 $f(x)$ 的次数为 n，记为 $\deg f=n$。

例 7.12　整系数多项式环 $Z[X]$ 中的 $3x+2$ 的次数为 1，x^2+2x+4 的次数为 2，x^3+1 的次数为 3。

像整数环一样，引入整除的概念到多项式环中。表 7.1 给出了概念之间的类比。

下面依次介绍这些概念和公式，可以通过上述类比进行知识迁移。

定义 7.12　设 $f(x)$，$g(x)$ 是整环 R 上的任意两个多项式，其中 $g(x)\ 0$，如果存在一个多项式 $q(x)$ 使得等式

表 7.2　多项式环与整数环中整除相关概念和方法间的对照

多项式环	整数环
不可约多项式	素数
可约多项式(合式)	合数
多项式 Euclid 除法	整数 Euclid 除法
因式(最大公因式)	因数(最大公因数)
倍式(最小公倍式)	倍数(最小公倍数)
不完全商	不完全商
余数	余式
不可约多项式判定检测 degf 为 $n/2$	素数判定检测上限为根号 n
$(f(x),g(x))=(g(x),h(x))$	$(a,b)=(b,r)$
$s(x)f(x)+t(x)g(x)=((g(x),h(x))$	$sa+tb=(a,b)$

$$f(x)=g(x)q(x)$$

成立，则称 $g(x)$整除 $f(x)$，或者 $f(x)$被 $g(x)$整除，记作 $g(x)\mid f(x)$。这时，$g(x)$叫做 $f(x)$的因式，$f(x)$叫做 $g(x)$的倍式。否则，称 $g(x)$不能整除 $f(x)$，或者 $f(x)$不能被 $g(x)$整除。

设 $f(x)$，$g(x)$，$h(x)$是整环 R 上的多项式，满足 $f(x)g(x)=h(x)$，则有

$$\deg f+\deg g=\deg h$$

例 7.13　整系数多项式环 $Z[X]$中，$2x+3\mid 2x^2+3x$，$x^2+1\mid x^4-1$。

定义 7.13　设 $f(x)$是整环 R 上的非常数多项式，如果除了显式因式 1 和 $f(x)$外，$f(x)$没有其他因式，则 $f(x)$叫做不可约多项式，否则，$f(x)$叫做可约多项式，或者合式。

注意，多项式是否可约与其所在的环或者域有关。也就是说，具体可约不可约与系数所在的群有关。

例 7.14　多项式 x^2+1 在 $Z[X]$中是不可约的，但是在 $F_2[X]$中是可约的 $x^2+1=(x+1)^2$，在复数域 C 上也是也可约的，$x^2+1=(x+i)(x-i)$。

定理 7.7　给定域 K 上的 n 次多项式 $f(x)$，如果 $p(x)$是 $f(x)$的次数最小因式，则 $p(x)$是域 K 上的不可约多项式，且 $\deg p\leqslant 1/2\deg f$。

证明：反证法。如果 $p(x)$为可约多项式，则存在因式 $p_1(x)$，$\deg p_1<\deg p$，使得 $p_1(x)\mid p(x)$，从而 $p_1(x)\mid f(x)$，这与 $p(x)$是 $f(x)$的次数最小的因式矛盾，所以，$p(x)$是不可约多项式。

另外，因为 $f(x)$是可约多项式，所以存在多项式 $f_1(x)$，使得

$$f(x)=f_1(x)p(x),1\leqslant\deg p\leqslant\deg f_1<n$$

因此，$2\deg p\leqslant\deg f_1+\deg p=n$，于是，$\deg p\leqslant 1/2\deg f$。 ■

定理 7.8（多项式 Euclid 除法）　设

$$f(x)=a_nx^n+\cdots+a_1x+a_0$$
$$g(x)=b_nx^m+\cdots+b_1x+b_0$$

是整环 R 上的两个多项式，则一定存在多项式 $q(x)$ 和 $r(x)$，使得

$$f(x)=q(x)g(x)+r(x),\deg r(x)<\deg g(x)$$

证明： 对 $f(x)$ 的次数 $\deg f=n$ 做数学归纳法。

(1) 如果 $\deg f<\deg g$，则取 $q(x)=0, r(x)=f(x)$，结论成立。

(2) 设 $\deg f\geqslant\deg g$，假设结论对 $\deg f<n$ 的多项式成立。

对于 $\deg f=n\geqslant\deg g$，有

$$f(x)-a_nx^{n-m}g(x)=(a_{n-1}-a_nb_{m-1})x^{n-1}+\cdots+$$
$$(a_{n-m}-a_nb_0)x^{n-m}+a_{n-m-1}x^{n-m-1}+\cdots+a_0$$

这说明 $f(x)-a_nx^{n-m}g(x)$ 是次数 $\leqslant n-1$ 的多项式，对其运用归纳假设或情形(1)，存在整系数多项式 $q_1(x)$ 和 $r_1(x)$ 使得

$$f(x)-a_nx^{n-m}g(x)=q_1(x)g(x)+r_1(x)$$

因此，$q(x)=a_nx^{n-m}+q_1(x), r(x)=r_1(x)$ 为所求。

根据数学归纳法，结论成立。■

注意，多项式是否可约与其所在的环或者域有关类似，多项式 Euclid 除法需要考虑多项式所在的环或者域。

例 7.15 在素域 F_7 上，有 $x^3+x+1=(4x+5)(2x^2+x+1)+6x+3$。在有理数域 Q 上，有 $x^3+x+1=(1/2x-1/4)(2x^2+x+1)+3/4x+5/4$。

定义 7.14 上述定理中，$q(x)$ 叫做 $f(x)$ 被 $g(x)$ 除所得的不完全商，$r(x)$ 叫做 $f(x)$ 被 $g(x)$ 除所得的余式。

推论 1： 设 $f(x)=a_nx^n+\cdots+a_1x+a_0$ 是整环 R 上的多项式，$a\in R$，则一定存在多项式 $q(x)$ 和常数 $c=f(a)$，使得

$$f(x)=(x-a)q(x)+c$$

证明： 根据定理 7.8，对于 $f(x), g(x)=x-a\in R[x]$，存在多项式 $q(x), r(x)$，使得

$$f(x)=g(x)q(x)+r(x),\quad \deg r<\deg g$$

因为 $\deg g=1, \deg r<\deg g$，所以 $\deg r=0, r(x)=c\in R$。即有

$$f(x)=(x-a)q(x)+c$$

特别取 $x=a$，有 $c=f(a)$。■

推论 2： 设 $f(x)=a_nx^n+\cdots+a_1x+a_0$ 是整环 R 上的多项式，$a\in R$，则 $x-a\mid f(x)$ 的充要条件是 $f(a)=0$。

例 7.16 设 $f(x)=x^4+x^3+x+1, g(x)=x^2+x+1$ 是 $F_2[X]$ 中的多项式，求 $q(x)$ 和 $r(x)$ 使得

$$f(x)=g(x)q(x)+r(x),\deg r<\deg g$$

解答： 利用竖式除法，逐次消去最高次项

x^4+x^3+x+1	x^2+x+1
$x^4+x^3+x^2$	x^2+1
x^2+x+1	
x^2+x+1	
0	

$q(x)=x^2+x+1, r(x)=0$。

例 7.17 设 $f(x)=x^4+x^2+x+1, g(x)=x+1$ 是 $F_2[X]$中的多项式，求 $q_1(x)$和 $r_1(x)$，使得

$$f(x)=g(x)q_1(x)+r_1(x), \mathrm{deg} r_1<\mathrm{deg} g$$

解答：利用竖式除法，逐次消去最高次项

$$\begin{array}{r|l}
x^4+x^2+x+1 & x+1 \\
x^4+x^3 \quad\quad\quad & \overline{x^3+x^2+1} \\
\hline
x^3+x^2+x+1 & \\
x^3+x^2 \quad\quad\; & \\
\hline
x+1 & \\
x+1 & \\
\hline
0 \;\; &
\end{array}$$

$q_1(x)=x^3+x^2+1, r_1(x)=0$。

例 7.18 设 $f(x)=x^4+x+1, g(x)=x^2+1$ 是 $F_2[X]$中的多项式，求 $q_1(x)$和 $r_1(x)$使得

$$f(x)=g(x)q_1(x)+r_1(x), \mathrm{deg} r_1<\mathrm{deg} g$$

解答：利用竖式除法，逐次消去最高次项

$$\begin{array}{r|l}
x^4+x+1 \quad & x^2+1 \\
x^4+x^2 \quad\quad\; & \overline{x^2+1} \\
\hline
x^2+x+1 & \\
x^2+1 \quad & \\
\hline
x \;\; &
\end{array}$$

$q_1(x)=x^2+1, r_1(x)=x$。

根据多项式 Euclid 除法，可以用来判定多项式整除关系。

定理 7.9 设 $f(x), g(x)$是域 K 上两个多项式，则 $f(x)$被 $g(x)$整除的充要条件是 $f(x)$被 $g(x)$除的余式 $r(x)$为零多项式。

类似于素数的判定，可得如下判定方法：

定理 7.10 设 $f(x)$是域 K 上的 n 次多项式，如果对于所有的不可约多项式 $p(x)$，$\mathrm{deg}p\leqslant n/2$，都有 $p(x)$ $f(x)$，则 $f(x)$一定是一个不可约多项式。

例 7.19 证明 $f(x)=x^2+x+1$ 为 $F_2[X]$中的不可约多项式。

证明：$F_2[X]$中次数$\leqslant n/2=1$ 的不可约多项式有 $x, x+1$。将 $f(x)$和这些不可约多项式做 Euclid 除法，有

$$f(x)=x(x+1)+1$$
$$f(x)=(x+1)+1$$

都不能整除 $f(x)$，因此，$f(x)$为不可约多项式。

例 7.20 证明 $f(x)=x^3+x+1$ 是 $F_2[X]$中的不可约多项式。

证明：$F_2[X]$中次数$\leqslant n/2=1$ 的不可约多项式有 $x, x+1$。将 $f(x)$和这些不可约多项式做 Euclid 除法，有

$$f(x)=x(x^2+1)+1$$
$$f(x)=(x+1)^2x+1$$

都不能整除 $f(x)$，因此，$f(x)$为不可约多项式。

例 7.21　证明 $f(x)=x^3+x^2+1$ 是 $F_2[X]$中的不可约多项式。

证明：$F_2[X]$中次数$\leqslant n/2=1$ 的不可约多项式有 $x,x+1$。将 $f(x)$和这些不可约多项式做 Euclid 除法，有

$$f(x)=x^2(x+1)+1$$
$$f(x)=(x+1)x^2+1$$

都不能整除 $f(x)$，因此，$f(x)$为不可约多项式。

类似与整数中的最大公因数和最小公倍数，可以给出多项式环 $R[X]$中的最大公因式和最小公倍式。

设 $f(x),g(x)\in R[X]$，$d(x)\in R[X]$叫做 $f(x),g(x)$的最大公因式，如果：

(1) $d(x)\mid f(x),d(x)\mid g(x)$；

(2) 若 $h(x)\mid f(x),h(x)\mid g(x)$，则 $h(x)\mid d(x)$。

$f(x),g(x)$的最大公因式记作$(f(x),g(x))$。

考虑域 K 上的最大公因式时，约定其最高次项系数为 1，则最大公因式是唯一的。

$f(x)$和 $g(x)$叫作互素的，如果它们的最大公因式$(f(x),g(x))=1$。

设 $f(x),g(x)\in R[X]$，$D(x)\in R[X]$叫做 $f(x),g(x)$的最小公倍式，如果：

(1) $f(x)\mid D(x),g(x)\mid D(x)$；

(2) 若 $f(x)\mid h(x),g(x)\mid h(x)$，则 $D(x)\mid h(x)$。

$f(x),g(x)$的最小公倍式记作$[f(x),g(x)]$。

与整数除法有类似的如下递推关系。

定理 7.11　设 $f(x),g(x),h(x)$是域 K 上的三个非零多项式，如果

$$f(x)=q(x)g(x)+h(x)$$

其中 $q(x)$是域 K 上的多项式，则

$$(f(x),\ g(x))=(g(x),\ h(x))$$

证明：设 $d(x)=(f(x),g(x))$，$d'(x)=((g(x),h(x))$，则 $d(x)\mid f(x),d(x)\mid g(x)$，进而

$$d(x)\mid f(x)+(-q(x))g(x)=h(x)$$

因此，$d(x)$是 $g(x),h(x)$的公因式，$d(x)\mid d'(x)$。

同理，$d'(x)$是 $f(x),g(x)$的公因式，$d(x)\mid d'(x)$。

因此 $d(x)=d'(x)$。 ■

与整数除法一样利用反复 Euclid 除法，可以计算$(f(x),g(x))$。

设 $f(x),g(x)$是域 K 上的多项式，$\deg g\geqslant 1$。记 $r_{-2}(x)=f(x)$，$r_{-1}(x)=g(x)$，反复执行多项式 Euclid 除法，有：

$$
\begin{aligned}
r_{-2}(x) &= q_0(x)r_{-1}(x) + r_0(x) \quad 0 \leqslant \deg r_0 \leqslant \deg r_{-1} \\
r_{-1}(x) &= q_1(x)r_0(x) + r_1(x) \quad 0 \leqslant \deg r_1 \leqslant \deg r_0 \\
r_0(x) &= q_2(x)r_1(x) + r_2(x) \quad 0 \leqslant \deg r_2 \leqslant \deg r_1 \\
&\cdots \\
r_{k-4}(x) &= q_{k-2}(x)r_{k-3}(x) + r_{k-2}(x) \quad 0 \leqslant \deg r_{k-2} \leqslant \deg r_{k-3} \\
r_{k-3}(x) &= q_{k-1}(x)r_{k-2}(x) + r_{k-1}(x) \quad 0 \leqslant \deg r_{k-1} \leqslant \deg r_{k-2} \\
r_{k-2}(x) &= q_k(x)r_{k-1}(x) + r_k(x) \quad r_k(x) = 0
\end{aligned} \tag{7.1}
$$

经过有限步骤，必然存在 k 使得 $rk(x)=0$，这是因为

$$0 \leqslant \deg r_k < \deg r_{k-1} < \deg r_{k-2} < \cdots < \deg r_1 < \deg r_0 < \deg r_{-1} = \deg g$$

且 deg$g$ 是有限正整数。

定理 7.12 设 $f(x),g(x)$是域 K 上的多项式，$\deg g \geqslant 1$，则

$$(f(x),g(x))=r_{k-1}(x)$$

其中 $r_{k-1}(x)$是多项式 Euclid 除法中最后一个非零除式。

证明：根据定理 7.11，有

$$
\begin{aligned}
(f(x),g(x)) &= (r_{-2}(x),r_{-1}(x)) \\
&= (r_{-1}(x),r_0(x)) \\
&= (r_0(x),r_1(x)) \\
&= \cdots \\
&= (r_{k-2}(x),r_{k-1}(x)) \\
&= (r_{k-1}(x),0) \\
&= r_{k-1}(x)
\end{aligned}
$$

■

将上述过程反过来计算，则可以找到 $s(x),t(x)$，使得

$$s(x)f(x)+t(x)g(x)=(f(x),g(x))$$

例 7.22 $f(x)=x^7+x^5+x^2+1\in F_2[X]$，$g(x)=x^4+x^2+x\in F_2[X]$，求$(f(x),g(x))$，并求 $s(x),t(x)$，使得 $s(x)f(x)+t(x)g(x)=(f(x),g(x))$。

解答：利用多项式 Euclid 除法以及逆过程，有：

$$
\begin{aligned}
x^7+x^5+x^2+1 &= (x^3+1)(x^4+x^2+x)+(x+1) \\
x^4+x^2+x &= (x^3+x^2+1)(x+1)+1 \\
x+1 &= (x+1)1
\end{aligned}
$$

于是$(f(x),g(x))=1$。

反过来写：

$$
\begin{aligned}
1 &= x^4+x^2+x+(x^3+x^2+1)(x+1) \\
&= x^4+x^2+x+(x^3+x^2+1)(f(x)+(x^3+1)g(x)) \\
&= (x^3+x^2+1)f(x)+(x^3+x^2+1)(x^3+1)g(x) \\
&= (x^3+x^2+1)f(x)+(x^6+x^5+x^2+1)g(x)
\end{aligned}
$$

于是 $s(x)=x^3+x^2+1$，$t(x)=x^6+x^5+x^2+1$。

定理 7.13 设 $f(x),g(x)$是域 K 上的多项式，则

$$s_{k-1}(x)f(x)+t_{k-1}(x)g(x)=(f(x),g(x))$$

对于 $j=0,1,2,\cdots,k-1$，这里 s_j,t_j 归纳定义为

$$s_{-2}(x)=1,s_{-1}(x)=0,\ s_j(x)=s_{j-2}(x)-q_j(x)s_{j-1}(x),$$
$$t_{-2}(x)=1,t_{-1}(x)=0,\ t_j(x)=t_{j-2}(x)-q_j(x)t_{j-1}(x),$$

$j=0,1,2,\cdots k-1$。

其中 $q_j(x)$ 是(7.1)式中的不完全商。

证明和整数的情况类似。

由定理 7.13 可得到多项式扩展的 Euclid 除法算法 7.1，该算法在计算最大公因式的同时，也计算出模不可约多项式的乘法逆元。

算法 7.1　多项式 Extended Euclid 算法求逆元。

输入：两个多项式 m(x)，b(x)，degb＜degm，m(x)是不可约多项式。

输出：(m(x)，b(x))，b(x)在模 m(x)中的乘法逆元。

```
ExtendedEuclid(m(x), b(x)){
1 (A1(x),A2(x),A3(x))← (1,0,m(x));(B1,B2,B3)← (0,1,b(x));
2 If B3(x)=0   Return A3(x), 'No Inverse';
3 If B3(x)=1   Return B3(x), B2(x);
4 Q←A3(x)/B3(x);
5 (Temp1(x),Temp2(x),Temp3(x)) ← (A1(x)-Q(x)B1(x),A2(x)-Q(x)B2(x),A3(x)-Q(x)B3(x))
6 (A1(x),A2(x),A3(x))← (B1(x),B2(x),B3(x))
7 (B1(x),B2(x),B3(x))← (Temp1(x),Temp2(x),Temp3(x))
8 Goto 2
}
```

容易看到，算法 7.1 和算法 1.4 是类似的。

7.2　域

定义 7.15　给定 $R[X]$ 中的一个首一多项式 $m(x)$，两个多项式 $f(x)$，$g(x)$ 模 $m(x)$ 同余，如果 $m(x)\mid f(x)-g(x)$，记作 $f(x)\equiv g(x)\pmod{m(x)}$；否则叫做模 $m(x)$ 不同余，记作

$$f(x)\not\equiv g(x)\pmod{m(x)}$$

任一多项式 $f(x)$ 都与其被 $m(x)$ 除的余式 $r(x)$ 模 $m(x)$ 同余，余式 $r(x)$ 叫做 $f(x)$ 模 $m(x)$ 的最小余式，记作 $(f(x)\bmod m(x))$。

定理 7.14　设 $f(x)\in F[X]$，则 $F[X]/f(x)$ 为域当且仅当 $f(x)$ 为域 F 上的不可约多项式。

与 Z/nZ 类似，$F[X]/f(x)$ 中的元素即为次数小于 $f(x)$ 次数的所有 F 上的多项式，其中的加法、乘法运算分别为模多项式 $f(x)$ 的加法和乘法运算，若 $F=F_p$，$\deg f=n$，则

$|F_p[X]/f(x)|=p^n$。

例 7.23 设 $f(x)=x^2+x+1\in F_2[X]$，则 $f(x)$ 在 F_2 上是不可约的，由定理 7.14 知，$F_2[X]/f(x)$ 构成一个具有 $|F_2[X]/f(x)|=2^2=4$ 个元素的域。$|F_2[X]/f(x)|=\{[0],[1],[x],[x+1]\}$。其元素运算表如表 7.3 和表 7.4 所示。

表 7.3　$F_2[X]/f(x)$ 的加法表

$+$	$[0]$	$[1]$	$[x]$	$[x+1]$
$[0]$	$[0]$	$[1]$	$[x]$	$[x+1]$
$[1]$	$[1]$	$[0]$	$[x+1]$	$[x]$
$[x]$	$[x]$	$[x+1]$	$[0]$	$[1]$
$[x+1]$	$[x+1]$	$[x]$	$[1]$	$[0]$

表 7.4　$F_2[X]/f(x)$ 的乘法表

$\cdot$	$[0]$	$[1]$	$[x]$	$[x+1]$
$[0]$	$[0]$	$[0]$	$[0]$	$[0]$
$[1]$	$[0]$	$[1]$	$[x]$	$[x+1]$
$[x]$	$[0]$	$[x]$	$[x+1]$	$[1]$
$[x+1]$	$[0]$	$[x+1]$	$[1]$	$[x]$

从运算表可以看出，$[0]$是零元，$[1]$是单位元。

例 7.24 设 $F=Z/pZ$ 是一个有限域，其中 p 为素数，$p(x)$是 $F[X]$中的 n 次不可约多项式，则

$$F[X]/(p(x))=\{a_nx^n+\cdots+a_1x+a_0\,|\,a_0\in F\}$$

记为 F_{p^n}。这个域中元素的个数为 p^n。

F_{p^n} 中的加法和乘法分别是：

$$f(x)+g(x)=((f+g)(x)\ (\mathrm{mod}\ p(x))$$

$$f(x)\cdot g(x)\ =\ ((fg)(x)\ (\mathrm{mod}\ p(x))$$

设 F_q是 q 元有限域，其特征 p 为素数。下面证明：$F_q^*=F_q\backslash\{0\}$是 $q-1$ 阶循环乘群。下面讨论 F_q^* 的一些性质。

定理 7.15 F_q^* 的任意元素的阶整除 $q-1$。

定义 7.16 有限域 F_q的元素 g 叫做生成元，如果它是 F_q^* 的生成元，即阶为 $q-1$ 的元素。当 g 是 F_q的生成元时，有 $F_q=\{0,g^0=1,g,g^2,\cdots,g^{q-2}\}$。

类似第 5 章中原根的判定方法，可以得到有限域 F_q 的元素 g 是否为生成元的一个判定方法。

定理 7.16 设 g 是有限域 F_{p^n} 中的元素，p^n-1 的所有不同素因数是 $q_1,\cdots,q_k$，则 g 是有限域 F_{p^n} 的一个生成元的充要条件是

$$g^{(p^n-1)/q_i}\neq 1,i=1,\cdots,k$$

定理 7.16 其实给出了求有限域的生成元的方法。

下面给出有限域的三条结构定理。

定理 7.17　设 F 是一个特征为素数 p 的有限域，则 F 中的元素个数为 p^n，n 是一个正整数。

定理 7.18　（存在性）对于任何素数 p 和任意正整数 n，都存在一个有限域含有 p^n 个元素。

定理 7.19　（唯一性）任意两个 $q=p^n$ 元域都同构，即 p^n 元域在同构意义下是唯一的。

例 7.25　Z_p 是一个特征为素数 p 的域，且其非零元素 $Z_p*=\{1, 2, \cdots, p-1\}$ 形成一个 $p-1$ 阶循环群。Z_p 不是特征为 p 的唯一域（见定理 7.17）。另外事实上，任何有限域的乘法群都是循环群，例如 F_{q*}（这里 $q=p^n$）一个 p^n-1 阶循环群。

例 7.26　$Z_p[X]$ 中有许多次数为 n 的不可约多项式，但可以证明，有任何两个 n 次不可约多项式构造的域是同构的。因此，存在唯一的 p^n（p 是素数，$n\geqslant 1$）个元素的有限域，记为 F_{q*}（这里 $q=p^n$）。n 为 1 是，F_p 与 Z_p 相同。可以证明，如果存在 r 个元素的有限域，则一定存在某个素数 p 及某个整数 $n\geqslant 1$，使得 $r=p^n$（见定理 7.18 和定理 7.19。）

定理 7.20　每个有限域都有生成元，如果 g 是 F_q 的生成元，则 g^d 是 F_q 的生成元当且仅当 d 和 $q-1$ 的最大公因数 $(d, q-1)=1$，特别地，F_q 有 $\Phi(q-1)$ 个生成元。

推论　设 $q=p^n$，p 为素数，$d|q-1$，则有限域 F_q 中有阶为 d 的元素。

例 7.27　求 $F_{2^4}=F_2[X]/(x^4+x+1)$ 的生成元 $g(x)$，并计算 $g(x)^t$，$t=1,2,\cdots,14$ 和所有的生成元。

解答：因为 $|F_{2^4}^*|=15=3\cdot 5$，所以满足

$$g(x)^3\not\equiv 1 \pmod{x^4+x+1},\ g(x)^5\not\equiv 1 \pmod{x^4+x+1}$$

的元素 $g(x)$ 都是生成元。

对于 $g(x)=x$，有

$$x^3\not\equiv 1 \pmod{x^4+x+1},\ x^5\not\equiv 1 \pmod{x^4+x+1}$$

所以 $g(x)=x$ 是 $F_2[X]/(\text{mod } x^4+x+1)$ 的生成元。

对于 $t=0,1,\cdots,14$，计算 $g(x)^t \pmod{x^4+x+1}$ 如下：

$$g(x)^0=1, g(x)^1=x, g(x)^2=x^2, g(x)^3=x^3$$
$$g(x)^4=x+1, g(x)^5=x^2+x, g(x)^6=x^3+x^2, g(x)^7=x^3+x+1$$
$$g(x)^8=x^2+1, g(x)^9=x^3+x, g(x)^{10}=x^2+x+1, g(x)^{11}=x^3+x^2+x$$
$$g(x)^{12}=x^3+x^2+x+1,\ g(x)^{13}=x^3+x^2+1, g(x)^{14}=x^3+1$$

所有生成元 $g(x)^t$，$(t,15)=1$，取 $t=1,2,4,7,8,11,13,14$ 时的计算结果可得到生成元。

有限域在对称加密算法设计中有大量的应用，例如在 AES 密码算法中便涉及有限域 $F_{2^8}=F_2[X]/(x^8+x^4+x^3+x+1)$ 及其生成元 $g=x+1$，7.3.2 节将详述。

7.3 环和域在 AES 加密中的应用

1997 年美国 NIST 发起公开征集高级加密标准(Advanced Encryption Standard, AES)算法的活动,目的是寻找一个安全性能更好的分组密码算法替代 DES。AES 的基本要求是安全性能不能低于三重 DES,且执行速度比三重 DES 快。而且分组长度为 128 位,并能支持长度为 128 位、192 位、256 位的密钥。

1998 年,NIST 召开了第一次 AES 候选会议,公布了 15 个满足 AES 基本要求的算法作为候选算法,并提请公众协助分析这些候选算法。1999 年 NIST 召开了第二次 AES 获选会议,公布了第一阶段的分析和测试结果,从 15 个候选算法中选出了 5 个决赛算法(Mars、RC6、Rijindael、Serpent 和 Twofish)。2000 年,NIST 召开第三次 AES 候选会议,通过对决赛算法的安全性、速度以及通用性等要素的综合评估,最终决定比利时密码学家 Joan Daemen 和 Vincent Rijmen 提出的"Rijndael"数据加密算法修改后作为 AES。2001 年 NIST 正式公布 AES,并与 2002 年 5 月开始生效。

7.3.1 AES 的设计思想

1. 基本安全参数

分组长度为 128 位,密钥长度可以独立设置为 128、192 和 256 位,因此 AES 有 3 个版本,AES-128、AES-192、AES-256。相应的迭代轮数为 10、12、14。

128 位的输入明文分组为 16 字节,通常用图形表示为 4 • 4 的正方形矩阵,称为状态(State)矩阵。例如,对于 128 比特的分组,可分成 16 个字节,从左到右为 $s_{00}\,s_{10}\,s_{20}\,s_{30}\,s_{01}\,s_{11}\,s_{21}\,s_{31}\,s_{02}\,s_{12}\,s_{22}\,s_{32}\,s_{03}\,s_{13}\,s_{23}\,s_{33}$。状态矩阵 $\boldsymbol{S}$ 表示为如下矩阵:

$$\boldsymbol{S}=\begin{bmatrix} s_{00} & s_{01} & s_{02} & s_{03} \\ s_{10} & s_{11} & s_{12} & s_{13} \\ s_{20} & s_{21} & s_{22} & s_{23} \\ s_{30} & s_{31} & s_{32} & s_{33} \end{bmatrix}$$

AES 算法的分组长度固定为 128 比特。Rijndael 中分组长度还可为 192 或 256 比特,则相应的列数为 6 列和 8 列。因此,AES 可视为 Rijndael 算法的子集。

类似地,可从输入密钥构造轮密钥 RoundKey 矩阵。

2 设计思想

Rijndael 的设计目标是:抵抗所有当时已知的攻击;在多个平台上速度快,编码紧凑;设计简单。Rijndael 没有采用 DES 中的 Feistel 结构,其轮函数是 3 个不同的可逆变换组成的。轮函数中有三种功能层:

(1) 线性混合层——确保多轮之后的扩散;

(2) 非线性层——将具有最优的"最坏情况非线性特性"的 S 盒并行使用;

(3) 密钥加层——将轮密钥和每一轮结果进行相加(异或)。

7.3.2　AES 中 S 盒的设计

1. AES 的总体结构

如图 7.3 所示，Rijndael 加密算法的轮函数采用 SP(Substitution-Permutation)结构，每一轮由字节代换(SubByte)、行移位变换(ShiftRow)、列混合变换(MixColumn)、轮密钥加变换(AddRoundKey)组成。加密过程执行一个"初始轮密钥加"，然后执行 N_r-1 (N_r 为迭代轮数)次"中间轮变换"，以及一个"末轮变换"。

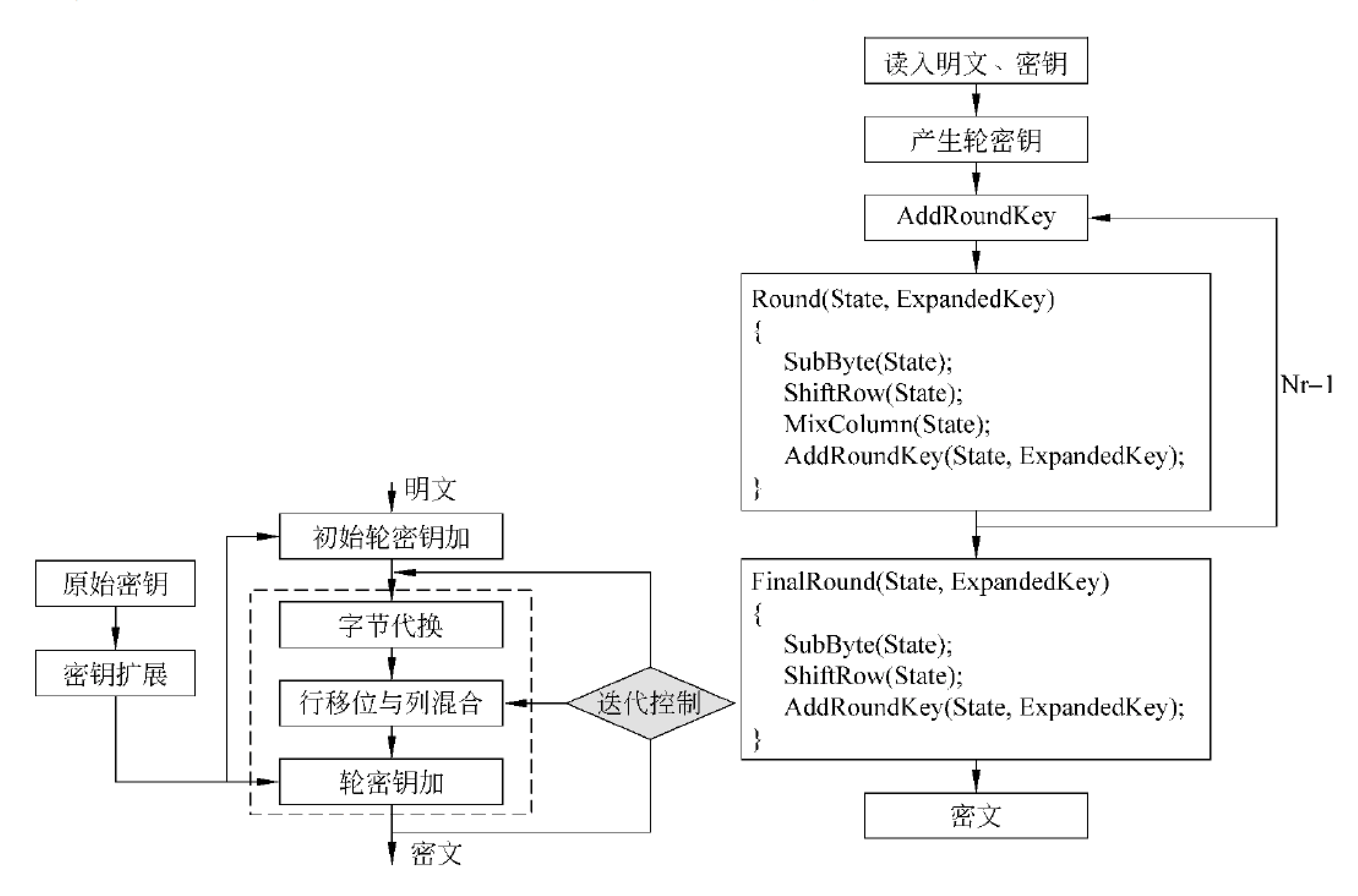


图 7.3　AES 总体结构

2. AES 中的字节代换(SubByte)部分

这是 AES 中的每一轮工作的首要部分。

通常将字节代换的计算结果先制成 S 盒表格，通过利用查表进行快速变换。

例如"F5"查表后得到"E6"，"56"查表后得到"B1"，等等。

例如，如下给出一个输入输出状态矩阵的例子：

F5	56	10	20		E6	B1	CA	B7
6B	44	57	39	S盒代换 →	7F	1B	5B	12
01	03	6C	21		7C	7B	50	FD
AF	30	32	34		79	04	23	18

问题是 S 盒是如何设计的？

AES 的 S 盒设计不像 DES 的 S 盒设计那么神秘(S 盒的设计方法没有给出具体原

因，因此有人担心其中会有后门)，而是有严格的数学计算。其设计原理是将一个字节非线性地变换为另一个字节。由两个变换复合而成：一个是求 $GF(2^8)$ 上的乘法逆，一个是仿射变换。

(1) 变换 1：求逆。

令字节 $z(x)=z_7x^7+z_6x^6+\cdots+z_1x+z_0$，先在有限域 $GF(2^8)$ 中求其关于 $m(x)=x^8+x^4+x^3+x+1$ 的乘法逆元，规定“00”的逆为“00”。

也就是说，将 $a\neq 0, a\in GF(2^8)$ 变换到其逆元。

即有映射：

$$t: GF(2^8)\to GF(2^8), a\mapsto t(a)$$
$$a=0, t(a)=0; a\neq 0, t(a)=a^{-1}$$

(2) 变换 2：仿射变换。

$$[y_0\ y_1\ y_2\ y_3\ y_4\ y_5\ y_6\ y_7]^T=\mathbf{A}[x_0\ x_1\ x_2\ x_3\ x_4\ x_5\ x_6\ x_7]^T\oplus[0\ 1\ 1\ 0\ 0\ 0\ 1\ 1]^T$$

这里 $\mathbf{A}$ 为一个规定的矩阵。

具体而言，给定 $u(x)=x^7+x^6+x^5+x^4+1, v(x)=x^7+x^6+x^2+x$，定义映射：

$$L_{u,v}: GF(2^8)\to GF(2^8), L_{u,v}(a)=b$$

对任意的

$$a=(a_7a_6a_5a_4a_3a_2a_1a_0)\in GF(2^8)$$

先将 a 表示成多项式

$$a(x)=a_7x^7+\cdots+a_2x^2+a_1x+a_0$$

然后计算

$$b(x)=u(x)a(x)+v(x)\ \mathrm{mod}\ (x^8+1)$$

设

$$b(x)=b_7x^7+\cdots+b_2x^2+b_1x+b_0$$

则

$$L_{u,v}(a)=b=[b_7b_6b_5b_4b_3b_2b_1b_0]\in GF(2^8)$$

仿射变换 L_{uv} 可用矩阵表示，即为

$$\begin{bmatrix}b_0\\b_1\\b_2\\b_3\\b_4\\b_5\\b_6\\b_7\end{bmatrix}=\begin{bmatrix}1&0&0&0&1&1&1&1\\1&1&0&0&0&1&1&1\\1&1&1&0&0&0&1&1\\1&1&1&1&0&0&0&1\\1&1&1&1&1&0&0&0\\0&1&1&1&1&1&0&0\\0&0&1&1&1&1&1&0\\0&0&0&1&1&1&1&1\end{bmatrix}\begin{bmatrix}a_0\\a_1\\a_2\\a_3\\a_4\\a_5\\a_6\\a_7\end{bmatrix}\oplus\begin{bmatrix}1\\1\\0\\0\\0\\1\\1\\0\end{bmatrix}$$

从另一个角度来看，满足：

$$b_i=a_i\oplus a_{(i+4)\ \mathrm{mod}\ 8}\oplus a_{(i+5)\ \mathrm{mod}\ 8}\oplus a_{(i+6)\ \mathrm{mod}\ 8}\oplus a_{(i+7)\ \mathrm{mod}\ 8}\oplus c_i$$

c_i 为 $(63)_8=(01100011)_2$ 的第 i 位。

可见，通过上述等式计算可快速进行，且具有混淆(Confusion)，分组加密通常由混淆

和置换两个部分组成的效果。

从一般原理上解释，SubByte 代换可视为 $S_{u,v}=L_{u,v}\cdot t$，即先做逆运算，再做仿射运算。虽然复合变换 $S_{u,v}$的两个变换 $L_{u,v}$，t 都是对 GF(2^8)中的元素进行计算，但是却使用了不同的数学结构：t 是在有限域 GF(2^8)$=F_2(x)/m(x)$上进行，而 $L_{u,v}$却是环 $F_2[x]/(x^8+1)$上进行。尽管 t 和 $L_{u,v}$都非常简单，但它们的复合运算却非常复杂。这种集合相同但数学结构不同的运算的复合，是 AES 的字节代换具有“非线性性”的保证。这一方法已成为分组密码设计的常用方法。同时，由于这两个变换都是可逆的，故存在逆复合变换。

例 7.28　以 F5 为例说明 S 盒的替代操作。不通过查表，而通过代数运算。首先求解“F5”在 GF(2^8)上的乘法逆元。输入“F5”对应“11110101”，对应多项式($x^7+x^6+x^5+x^4+x^2+1$)，求其模 $m(x)=x^8+x^4+x^3+x+1$ 的逆，即求($x^7+x^6+x^5+x^4+x^2+1$)$\cdot a(x)\equiv 1 \bmod m(x)$，通过扩展的 Euclidean 算法，求得其逆为(x^6+x^2+x)。表示为二进制为“01000110”。再进行仿射变换，代入矩阵

$$\begin{bmatrix}1&0&0&0&1&1&1&1\\1&1&0&0&0&1&1&1\\1&1&1&0&0&0&1&1\\1&1&1&1&0&0&0&1\\1&1&1&1&1&0&0&0\\0&1&1&1&1&1&0&0\\0&0&1&1&1&1&1&0\\0&0&0&1&1&1&1&1\end{bmatrix}\begin{bmatrix}0\\1\\1\\0\\0\\0\\1\\0\end{bmatrix}\oplus\begin{bmatrix}1\\1\\0\\0\\0\\1\\1\\0\end{bmatrix}=\begin{bmatrix}0\\1\\1\\0\\0\\1\\1\\1\end{bmatrix}$$

得到二进制结果为：111001110，对应十六进制结果为“E6”。

SubByte 用到了 AES 中的第一个基本运算，称为字节运算，即有限域 GF(2^8)上的运算（AES 的第二个基本运算是字运算，即系数在有限域 GF(2^8)上的运算）。$m(x)\in F_2[x]$是一个 8 次不可约多项式，故由 $m(x)$可生成一个有限域 GF(2^8)。

GF(2^8)$=F_2[x]/(m(x))=\{b_0+b_1x+b_2x^2+b_3x^3+b_4x^4+b_5x^5+b_6x^6+b_7x^7 \mid b_i\in F_2, i=0,1,\cdots,7\}=\{(b_7b_6b_5b_4b_3b_2b_1b_0)\mid b_i\in F_2, i=0,1,\cdots,7\}$

加法为模 2 加法，实际上相当于异或。减法其实等于加法，因为-1 的逆为 1。例如：$(x^6+x^4+x^2+x+1)+(x^7+x+1)=x^7+x^6+x^4+x^2$。多项式乘以 x(xtime 操作)，即左移 1 位。例如求 $z(x)\cdot x$，若 $z_7=0$，则结果为左移 1 位。若 $z_7=1$，则左移 1 位后，再求模，通常是减去模多项式 $m(x)$，减去即为加上。例如$(x^6+x^4+x^2+x+1)(x^7+x+1)=x^7+x^6+1 \bmod m(x)$。计算过程等同于计算$(57)_{16}\cdot(83)_{16}$，由于：

$$57_{16}\cdot 02_{16}=\text{xtime}(57_{16})=ae_{16}, 57_{16}\cdot 04_{16}=\text{xtime}(ae_{16})=47_{16}$$

$$57_{16}\cdot 08_{16}=\text{xtime}(47_{16})=8e_{16}, 57_{16}\cdot 10_{16}=\text{xtime}(8e_{16})=07_{16}$$

$$57_{16}\cdot 20_{16}=\text{xtime}(07_{16})=0e_{16}, 57_{16}\cdot 40_{16}=\text{xtime}(0e_{16})=1c_{16}$$

$$57_{16}\cdot 80_{16}=\text{xtime}(57_{16})=38_{16}$$

故$(57)_{16}\cdot(83)_{16}=57_{16}\cdot(01_{16}\oplus 02_{16}+80_{16})=(c1)_{16}=(1100\ 0001)_2$，即 x^7+x^6+1。

另外，由于 $m(x)$是不可约的，故可保证求出(需要加密的多项式)的逆元。

7.4 环在 NTRU 密码体制中的应用*

NTRU(Number Theory Research Unit)公开密钥算法是一种新的快速公开密钥体制，1996 年在 Crypto 会议上由布朗大学的 Hoffstein、Pipher、Silverman 三位数学家提出。经过几年的迅速发展与完善，该算法的密码学领域中受到了高度的重视并在实际应用(如嵌入式系统中的加密)中取得了很好的效果。

NTRU 是一种基于多项式环的密码系统，其加密、解密过程基于环上多项式代数运算和对数 p 和 q 的模约化运算，由正整数 N、p、q 以及 4 个 $N-1$ 次整系数多项式(f,g,r,m)集合来构建。N 一般为一个大素数，p 和 q 在 NTRU 中一般作为模数，这里不需要保证 p 和 q 都是素数，但是必须保证$(p,q)=1$，而且 q 比 p 要大得多。$R=Z[X]/(x^N-1)$ 为多项式环，其元素 $f(f\in R)$ 为：$f=a_{N-1}x^{N-1}+\cdots+a_1x+a_0$。定义 R 上多项式元素之间加运算为普通多项式之间的加运算，用符号＋表示。R 上多项式元素之间乘法运算为普通多项式的乘法运算，乘积结果要进行模多项式 x^N-1 的运算，用符号⊗表示。R 上多项式元素模 q 运算就是把多项式的系数作模 q 处理，用 mod q 表示。

NTRU 密码体制描述：

(1) **密钥生成**。随机选择两个 $N-1$ 次多项式 f 和 g 来生成密钥。利用扩展的 Euclidean 算法对 f 求逆。如果不能求出 f 的逆元，就重新选取多项式 f。用 F_p，F_q 表示 f 对 p 和 q 的乘逆。即：$F_q\otimes f\equiv 1 \bmod q$，$F_p\otimes f\equiv 1 \bmod p$。

计算：$h\equiv F_q\otimes g \bmod q$

最后得：公钥为(N,p,q,h)，私钥为(f,F_p)。

这里 F_p 可以从 f 容易地计算得到，但仍然作为私钥存储，这是因为在解密时需要使用这个多项式，而 F_p 和 q 就不需要存储了。

(2) **加密算法**。首先把消息表示成次数小于 N 且系数的绝对值至多为$(p-1)/2$ 的多项式 m，然后，随机选择多项式 $r\in L$，并计算：$c\equiv(pr\otimes h+m) \bmod q$。密文是多项式 c。

(3) **解密算法**。收到密文 c 后，可以使用私钥(f,F_p)对密文 c 进行解密。依次计算：

$$a\equiv(f\otimes c) \bmod q,\ a\in(-q/2,q/2)$$
$$b\equiv a \bmod p$$
$$m\equiv F_p\otimes b \bmod p$$

一致性证明：由于

$$\begin{aligned}a &\equiv f\otimes c \bmod q\equiv(f\otimes(\mathrm{pr}\otimes h+m) \bmod q) \bmod q\\ &\equiv(f\otimes \mathrm{pr}\otimes h+f\otimes m) \bmod q\\ &\equiv(f\otimes \mathrm{pr}\otimes F_q\otimes g+f\otimes m) \bmod q\\ &\equiv(\mathrm{pr}\otimes g+f\otimes m) \bmod q\end{aligned}$$

又因为 a 的系数在区间 $(-q/2,q/2)$，所以 $\mathrm{pr}\otimes g+f\otimes m$ 的系数在区间 $(-q/2,q/2)$，故 $\mathrm{pr}\otimes g+f\otimes m$ 模 q 后结果不变。因此

$$F_p\otimes b \bmod p \equiv (F_p\otimes a \bmod p) \bmod p \equiv F_p\otimes(\mathrm{pr}\otimes g+f\otimes m) \bmod p$$
$$\equiv (F_p\otimes \mathrm{pr}\otimes g+F_p\otimes f\otimes m) \bmod p \equiv m \bmod p$$

从而解密成功。

非正式地说，该加密算法的设计思路是：利用随机多项式 r 和公钥 h 生成一个"密钥多项式"，利用这个密钥多项式对 m 进行加密得到密文多项式。解密时利用多项式取模，约去随机多项式 r，利用多项式的逆，解出明文多项式。可见，同一个明文在不同的加密中会产生不同的密文。

例 7.29　设 $(N,p,q)=(5,3,6)$，以及 $f=x^4+x-1$ 和 $g=x^3-x$，求公钥私钥对以及描述加密解密过程。

解答：由于 $(x^4+x-1)\otimes(x^3+x^2-1)\equiv 1 \bmod 3$，故有 $F_q=x^3+x^2-1$，同理可求得 $F_q=x^3+x^2-1$。又由于 $h\equiv F_p\otimes g \bmod 16 \equiv -x^4-2x^3+2x^2+1$，所以公钥为 $(N,p,q,h)=(5,3,16,-x^4-2x^3+2x^2+1)$；私钥为 $(f,F_p)=(x^4+x-1,x^3+x^2-1)$。

加密过程：首先将消息 m 表示成多项式 $m=x^2-x+1$，然后选取多项式 $r=x-1$，则密文为：$c\equiv 3r\otimes h+m\equiv -3x^4+6x^3+7x^2-4x-5 \bmod 16$。

解密过程：首先计算 $a\equiv f\otimes c\equiv 4x^4-2x^3-5x^2+6x-2 \bmod 16$，计算 $F_p\otimes a\equiv x^2-x+1 \bmod 3$，这样就恢复了消息 m。

讨论：解密过程有时候可能无法恢复出正确的明文，因为：

在解密过程

$$a'\equiv(f\otimes c) \bmod q\equiv f\otimes(\mathrm{pr}\otimes h+m) \bmod q\equiv(\mathrm{pr}\otimes g+f\otimes m) \bmod q$$

中，如果多项式 $\mathrm{pr}\otimes g+f\otimes m$ 的系数不在区间 $(-q/2,q/2)$，则

$$f\otimes(\mathrm{pr}\otimes h+m) \bmod q\neq \mathrm{pr}\otimes g+f\otimes m$$

设 $f\otimes(\mathrm{pr}\otimes h+m)=\mathrm{pr}\otimes g+f\otimes m+\mathrm{qu}$，$u$ 为多项式，并且 u 的系数不全为 0，计算：

$$e'\equiv F_p\otimes a' \bmod p\equiv F_p\otimes(\mathrm{pr}\otimes g+f\otimes m+\mathrm{qu}) \bmod p$$
$$\equiv F_p\otimes \mathrm{pr}\otimes g+F_q\otimes f\otimes m+F_p\otimes \mathrm{qu} \bmod p$$

由于 p 和 q 互素，所以 $e'\equiv m+F_p\otimes \mathrm{qu} \bmod p\neq m$，所以解密失败。

通过选择恰当的参数 N、p、q 就能够避免以上错误，例如取 $(N,p,q)=(107,3,65)$ 和 $(N,p,q)=(503,3,256)$，实验表明解密错误的概率小于 $5\cdot 10^{-5}$，这就是通常能正确解密的原因。

安全性讨论：

NTRU 算法的安全性是基于在一个具有非常大维数的格(Lattice)中寻找最短向量(Shortest Vector Problem，SVP)问题是困难的。只要恰当地选择 NTRU 的参数，其安全性与 RSA，ECC 等加密算法是一样安全的。同时，NTRU 基于的困难问题没有量子算法可解，也称为后量子密码(Post-quantum Cryptograph)，或者量子免疫密码(Quantum immune Cryptography)。

思　考　题

[1] 程序实现多项式 Euclid 除法。

[2] 程序实现多项式广义 Euclid 除法。

[3] 证明 x^4+x^3+1 是 $F_2[x]$中的不可约多项式，从而 $F_2[x]/(x^4+x^3+1)$是一个 F_{2^4} 域；

[4] 求 $F_{2^4}=F_2[x]/(x^4+x+1)$的生成元 $g(x)$，计算 $g(x)^t$，$t=0,1,\cdots,14$ 和所有生成元。

[5] 编写程序输出 AES 的 S 盒。

[6] 编写程序实现 NTRU 密码体制。

第 8 章 素性检测

本章介绍素数的判定方法。重点是 Fermat 测试，难点是 Miller-Rabin 测试。

8.1 素数的一些性质

生成合适的 p 和 q 是 RSA 中密钥生成以及 RSA 安全性的关键。素数产生的办法是随机选择一个数，然后测试其是否为素数。这里随机选择比从一个固定的表中选择素数要更加安全。虽然在 2002 年，Agrawal、Kayal 和 Saxena 证明了存在一个素性判断的多项式时间的确定性算法，但是其效率不如概率判定算法。因此在实际应用中，素性检测仍然主要利用概率多项式时间算法。

目前最大的已知素数是梅森(Mersenne)素数 $2^{43112609}-1$(此数字位长度是 12978189。梅森数是指形如 2^n-1 的数，记为 M_n，如果一个梅森数是素数，那么它称为梅森素数。该书是第 47 个素数，记为 $M_{43112609}$)，它是在 2008 年 8 月 23 日由 GIMPS 发现。梅森数常用来检验素性检测算法的性能。

思考以下几个问题：

(1) 素数会不会用完，即素数是无穷的吗？

定理 1.5 已经证明了素数有无穷多个。

(2) 素数在自然数中占的比重如何，是否能够很快找到一个素数？

定理 8.1 素数定理：设 $\pi(x)$表示$\leqslant x$ 的素数的个数，则

$$\lim_{x\to\infty}\frac{\pi(x)}{x/\ln x}=1$$

x 充分大时，$\pi(x)\approx x/\ln x$。由素数定理知，对于正整数 N，不超过 N 的素数数目大约为 $N/\ln N$。即任意一个整数，它小于 N 且是素数的概率为 $1/\ln N$。假设生成长度为 512bit 的素数(首位为 1)，则长度为 512bit 的素数个数为：

$$\begin{aligned}2^{513}/\ln 2^{513}-2^{512}/\ln 2^{512}&=2\cdot 2^{512}/(513\ln 2)-2^{512}/(512\cdot\ln 2)\\&\approx 0.0028\cdot 2^{512}\approx 2^{512}/357\approx 2^{503.5}\approx 10^{151}\end{aligned}$$

(这个数非常大，要知道宇宙中原子的数量也仅仅为 10^{77}。)任何一个长度为 512bit 的数为素数的概率为 $2^{503.5}/(2^{513}-2^{512})\approx 1/357$。即平均每 357 个(长度为 512bit)的数中就有一个是素数。其中偶数(末尾为二进制的 10)的数不用测试，于是平均测试为 $357/2\approx 179$ 次，即为了发现一个长度为 512bit 的素数，平均测试 179 个长度为 512bit 的数就会发现一个。

一般地，长度为 t 比特的素数大约有：

$$\frac{2^{t+1}}{\ln 2^{t+1}}-\frac{2t}{\ln 2^{t}}=\frac{2^{t}(t-1)}{t(t+1)\ln 2}$$

(3) 如果很多人都选择 512 比特的 p，会不会出现重复？

重复的概率可忽略，因为共有 10^{151} 个数，重复的概率为 $\sqrt{10^{151}}\approx 10^{75.5}$（即随机碰撞的概率，所谓的“生日攻击”），这个数非常大。

(4) 为什么不建立素数的数据库，利用搜索数据库来尝试分解因子？

将所有 512bit 的素数保存起来，需要 $512.2^{503.5}=2^{511.5}$bit，如果将 10GB($2^{36.3}$bit)保存在 1g 重的存储设备上，需要的存储设备的总重量为 $2^{455}\times 10^{3}$kg，这一重量是不可能实现的，将导致系统崩溃，进入黑洞。

下面介绍素性检测的方法。

一个平凡的方法就是检测任何不大于$\sqrt{n}$的素数是否能整除 n 来确定 n 的素性。显然在实际中需要更高效的算法。

8.2 Fermat 测试

Fermat 测试的依据是 Fermat 定理。

首先回顾一下 Fermat 定理：若 n 是素数，且 a 是任何满足 $1\leqslant a\leqslant n-1$ 的整数，则

$$a^{n-1}\equiv 1 \bmod n$$

因此，给定需要判定素性的数 n，若在区间内能找到一个整数 a 使得 $a^{n-1}\neq 1 \bmod n$，则足以说明 n 是一个合数。

算法 8.1 Fermat(n)。

```
/* Fermat 素性测试，测试 n           */
/* 输入：奇数 n≥3                    */
/* 输出：测试结果，素数或者合数      */
{
    随机选择整数 a，2≤a≤n-2。
    r= a^(n-1) mod n
    If r≠1 Return ("Composite")
    Return("Prime")
}
```

算法 8.1 的输出为“合数”，则一定为合数，若输出为数“素数”，则可能为素数。但是存在这样的数 n，对所有满足 $\gcd(a,n)$ 的整数 a，均有 $a^{n-1}\equiv 1 \bmod n$，称这样的 n 为 Carmichael 数，有研究表明这种数较稀少。

研究表明，如果算法输出为“Prime”（即通过了素性检测），则 n 确为素数的可能性大于 $1-\frac{1}{2}$。如果运行 t 次，算法均输出“Prime”，则 n 确为素数的可能性大于 $1-\frac{1}{2^{t}}$。

8.3　Solovay-Strassen 测试

Solovay-Strassen 测试是随着 RSA 密码体制而出现并广泛使用的第一种素性测试。它的原理是基于 Euler 准则。

这里给出 Legendre 符号的推广，即不再限定 p 为素数。

定义 8.1　假定 n 是一个奇正整数，且 n 的素数幂因子分解为 $n=\prod_{i=1}^{k} p_i^{e_i}$。设 a 为一个整数，那么雅克比(Jacobi)符号 $\left(\frac{a}{n}\right)$ 定义为：

$$\left(\frac{a}{n}\right)=\prod_{i=1}^{k}\left(\frac{a}{p_i}\right)^{e_i}。$$

可见，当 n 为素数时，Jacobi 符号就是 Legendre 符号。

但是，注意 Jacobi 符号 $\left(\frac{a}{n}\right)$ 等于 1，不能推出 a 是模 n 的平方剩余（相比 Legendre 符号，$\left(\frac{a}{p}\right)$ 等于 1，可以推出 a 是模 p 的二次剩余）。

例 8.1　计算 Jacobi 符号 $\left(\frac{6278}{9975}\right)$。

$$\begin{aligned}\left(\frac{6278}{9975}\right)&=\left(\frac{6278}{3}\right)\left(\frac{6278}{5}\right)^2\left(\frac{6278}{7}\right)\left(\frac{6278}{19}\right)\\&=\left(\frac{2}{3}\right)\left(\frac{3}{5}\right)^2\left(\frac{6}{7}\right)\left(\frac{8}{19}\right)\\&=(-1)(-1)^2(-1)(-1)\\&=-1\end{aligned}$$

也就是说，如果知道了 n 的因子分解，即可根据因子分解的结果，分别去求 Legendre 符号（即利用平方乘算法求幂）。

但是，在不知道因子分解的情况下，是否可以求出 Jacobi 符号？幸运的是，存在这样的多项式算法。该算法利用了 Jocobi 符号的性质，特别是利用了二次互反律。

Jacobi 符号的性质：

(1) 如果 n 是一个正奇数，且 $m_1\equiv m_2 \bmod n$，则 $\left(\frac{m_1}{n}\right)=\left(\frac{m_2}{n}\right)$。

(2) 如果 n 是一个正奇数，那么 $\left(\frac{m_1m_2}{n}\right)=\left(\frac{m_1}{n}\right)\left(\frac{m_2}{n}\right)$。

特别地，若 $m=2^kt$，t 为一个奇数，则 $\left(\frac{m}{n}\right)=\left(\frac{2}{n}\right)^k\left(\frac{t}{n}\right)$。

(3) 如果 n 是一个正奇数，那么 $\left(\frac{2}{n}\right)=\begin{cases}1, & n\equiv\pm1 \bmod 8\\-1, & n\equiv\pm3 \bmod 8\end{cases}$。

(4) 如果 m 和 n 是正奇数，则 $\left(\frac{m}{n}\right)=\begin{cases}-\left(\frac{n}{m}\right), & m\equiv n\equiv3 \bmod 4\\\left(\frac{n}{m}\right), & 其他\end{cases}$。

容易观察到，Jocobi 符号和 Legendre 符号的计算规则是相似的。

这里形象地解释一下，主要是便于初学者记忆和理解。非正式地说，把幂视为“分子”，模视为“分母”，则：

(1) 性质 1 用于将大分子转化为小“分子”(计算较大数的 Jacobi 符号转化为计算较小数的 Jacobi 符号)。

(2) 性质 2 把计算偶数“分子”转化为计算奇数“分子”(偶数的 Jacobi 符号转化为计算奇数的 Jacobi 符号)。

(3) 性质 3 用于计算$\left(\frac{2}{n}\right)$。

(4) 性质 4 最重要，称为二次互反律，就是交换“分子”和“分母”的位置，在“分母”比“分子”大时使用。

总之，性质 1,2,4 是为了化简，性质 3 为了求值。

使用的顺序一般是性质 2-3，或者性质 4-1-2-3。序列 2-3 指先考查“分子”是否为偶数，若是则利用性质 2，然后利用性质 3 求值。如“分子”不是偶数，则使用序列 4-1-2-3，即利用性质 4 使“分母”变小，性质 1 和性质 2 使“分子”变小后用性质 3 求值。

例 8.2 计算上例中的 Jacobi 符号。

$$
\begin{aligned}
\left(\frac{6278}{9975}\right)&=\left(\frac{2}{9975}\right)\left(\frac{3139}{9975}\right)=\left(\frac{3139}{9975}\right) && \text{性质 2-3}\\
&=-\left(\frac{9975}{3139}\right)=-\left(\frac{558}{3139}\right) && \text{性质 4-1}\\
&=-\left(\frac{2}{3139}\right)\left(\frac{279}{3139}\right)=\left(\frac{279}{3139}\right) && \text{性质 2-3}\\
&=-\left(\frac{3139}{279}\right)=-\left(\frac{70}{279}\right) && \text{性质 4-1}\\
&=-\left(\frac{2}{279}\right)\left(\frac{35}{279}\right)=-\left(\frac{35}{279}\right) && \text{性质 2-3}\\
&=\left(\frac{279}{35}\right)=\left(\frac{34}{35}\right) && \text{性质 4-1}\\
&=\left(\frac{2}{35}\right)\left(\frac{17}{35}\right)=-\left(\frac{17}{35}\right) && \text{性质 2-3}\\
&=-\left(\frac{35}{17}\right)=-\left(\frac{1}{17}\right) && \text{性质 4-1}\\
&=-1 && \text{Legendre 符号的含义}
\end{aligned}
$$

归纳这 4 个性质，可知主要的运算是性质 2 中的取模运算，性质 3 中的 2 的幂次分解运算，最终的运算为若 n 为奇数，$a=2^e a_1$，其中 a_1 为奇数，有

$$\left(\frac{a}{n}\right)=\left(\frac{2^e}{n}\right)\left(\frac{a_1}{n}\right)=\left(\frac{2}{n}\right)^e\left(\frac{n \bmod a_1}{a_1}\right)(-1)^{(a_1-1)(n-1)/4}$$

于是可以设计一个计算$\left(\frac{a}{n}\right)$的算法，该算法无须分解 n 的因子，由于在计算中使用了二次互反律，故在程序中使用递归实现更加容易。基本思想是 JACOBI$(a,n)=s\cdot$ JACOBI$(n \bmod a_1, a_1)$。

算法 8.2　JACOBI(a, n)。

```
/* Jacobi符号(Legendre符号)的计算 */
/* 输入：奇数 n≥3，整数 a,0≤a≤n-1。
/* 输出：Jacobi符号(a/n)(当 n 为素数时，等同于 Legendre符号)
{
If (a=0) Return(0);
If (a=1) Return(1);
将 a表示成 2^e a1                //a的 2的幂次分解运算，其中 a1 为奇数
If (IsEvenNumber(e) s←1;         //性质 2，幂指数 e为偶数，故幂总为 1
Else

  {
  If (n=±1 mod 8) s←1;
  If (n=±3 mod 8) s←-1;
  }                              //性质 3
If (n=3 mod 4) AND (a1=3 mod 4) s←-s;
n1←n mod a1;
If (a1=1) Return(s);
Else
Return(s·JACOBI(n1,a1));         //递归调用，性质 4
}
```

算法的时间复杂度为 $O((\log_2 n)^2)$。

容易看到，这与算法 4.4 是一样的。

到此，可以给出 Solovay-Strassen 测试算法，该测试中分别计算 $x \leftarrow \left(\frac{a}{n}\right)$ 和 $y \leftarrow a^{(n-1)/2} \bmod n$，如果 $x=y \bmod n$，则输出素数。否则，输出合数。合数的结论总是正确的(根据 Euler 准则)，但输出素数时不一定是素数。

算法 8.3　Solovay-Strassen(n)。

```
/* 测试 n是否为素数 */
/* 输入：奇数 n */
/* 输出："Prime"或者"Composite" */
{
随机选取整数 a，使得 a∈[1,n-1];
x←JACOBI(a,n);           //调用子函数
If (x=0) Return ("Composite");
y←a^((n-1)/2) mod n;
If (x=y mod n) Return ("Prime");
Else Return("Composite");
}
```

8.4 Miller-Rabin 测试*

实际中，通常使用的概率素性测试方法是 Miller-Rabin 测试，也称为强伪素性测试。该测试基于以下思想：

回顾 Fermat 测试，需要计算 $a^{n-1} \bmod n$。若结果为 1 则输出“素数”，不为 1 则输出“合数”。一个自然的想法是能不能少计算一点。

下面分析这种可能性：由于 n 是奇数，故 $n-1$ 为偶数，不妨设为 $n-1=2t$，于是 $a^{n-1}=(a^t)^2$，考查如果 $a^t=\pm1 \bmod n$，就直接输出“素数”，没有必要再继续计算了(因为平方后会得到 $a^{n-1}=1 \bmod n$)。这样就节省了计算的时间。

进一步而言，如果 $n-1=2^k t(k\geqslant0)$，则

$$a^{n-1}=(a^t)^{2^k}=(((a^t)\underbrace{^2)^2)\cdots)^2}_{k\text{次}}$$

同前面的分析，如果 $a^t=\pm1 \bmod n$，则必然有 $a^{n-1}=1 \bmod n$，直接输出“素数”，不用继续计算。否则，如果在 $k-1$ 次平方的过程中，有一个为 -1，则最终也有 $a^{n-1}=1 \bmod n$，输出“素数”，依次计算 $a^t,(a^t)^2,\cdots,(a^t)^{k-1}$，那么最终使得 $a^{n-1}=1 \bmod n$，输出“素数”的计算序列要么是：

$$\begin{array}{cccc} a^t, & (a^t)^2, & \cdots & (a^t)^{2^{k-1}} \\ \pm1, & 1, & \cdots & 1 \end{array}$$

或者

$$\begin{array}{ccccccc} a^t, & (a^t)^2, \cdots (a^t)^{2^{i-1}}, & (a^t)^{2^i}, & (a^t)^{2^{i+1}}, & \cdots, & (a^t)^{2^{k-1}} \\ \neq\pm1, & \neq\pm1,\cdots \neq\pm1, & -1, & 1, & \cdots, & 1 \end{array}$$

即在第 $i(i=0,1,\cdots,k-1)$ 次平方后，等于 -1。除此以外，均输出“合数”。

有读者会疑问如果第 i 次平方后等于 1，不也可以使得最后有 $a^{n-1}=1 \bmod n$ 吗？是的，但是这情况下，$1 \bmod n$ 的平方根不是 ±1，说明 n 不是素数。因此，只有两种情况。要么开始为 ±1，要么在第 $i(i=0,1,\cdots,k-1)$ 次平方计算的过程中出现 -1 即可。

根据上面的分析，可以得到算法如下：

算法 8.4 Miller-Rabin(n)

```
/* Miller-Rabin素性测试 */
/* 输入奇数 n≥3 */
/* 输出对 n素性的判断 */
{ 把 n-1写成 n-1=2^k t,其中 t是一个奇数
  随机选取整数 a,使得 1≤a≤n-1
  b←a^t mod n;
  If b≡1 mod n Return ("Prime");
  For i=0 to k-1
  {
  If b≡-1 mod n Return ("Prime");
```

```
    Else
    b←b² mod n;
    }
    Return("Composite");
}
```

可以证明 Miller-Rabin 算法的错误概率，即输出“素数”，但其实是合数的概率为1/4，这里不再给出证明。

比较 Fermat 测试，Solovay-Strassen 测试和 Miller-Rabin 测试三者的精确度，有如图 8.1所示的关系。

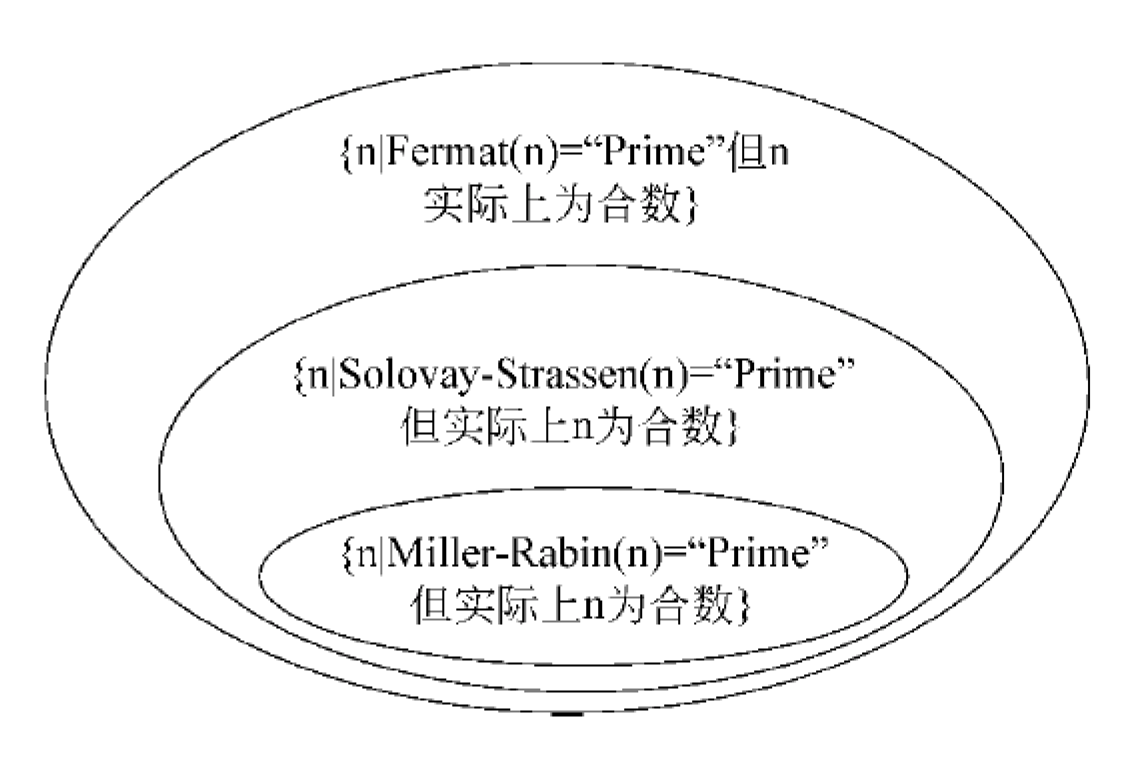


图 8.1　三种测试精确度的比较

可见 Miller-Rabin 精确度最高，错误概率最小。同时计算上也比 Solovay-Strassen 测试简单（Solovay-Strassen 测试需要计算 Jacobi 符号）。

思　考　题

[1] 实现 Miller-Rabin 素数检测算法，测试并找到一个对你而言最大的梅森素数。

[2] 编写 Fermat 测试、Solovay-Strassen 测试、Miller-Rabin 测试，比较它们的效率。

[3] 编写程序产生 300 十进制位左右的大素数。

高　级　篇

第 9 章 椭圆曲线群

本章介绍在椭圆曲线上构造的群。并基于这种群构造 DH 密钥协商协议，ElGamal 加密。

本章的重点是椭圆曲线群的构造。难点是 ElGamal 椭圆曲线加密方法和椭圆曲线快速标量点乘算法。

9.1 椭圆曲线群的概念

1. 椭圆曲线的定义

所谓椭圆曲线是指由韦尔斯特拉(Weierstrass)方程：

$$E: y^2 + axy + by = x^3 + cx^2 + dx + c$$

所确定的平面，其中 a,b,c,d 和 e 属于 F，F 是一个域，可以是有理数域、复数域或有限域，密码学中通常采用有限域。椭圆曲线是其上所有点(x,y)的集合，外加一个无限远点 O(该点不在椭圆曲线 E 上，记为 O，称此点为无限远点)。

椭圆曲线上可以提供无数的有限 Abel 群，具有丰富的群结构，不同的参数，得到不同的曲线，形成不同的群。而且，易于计算，从而可用来构造密码算法。在密码学中，常采用下列形式的椭圆曲线：

$$E: y^2 = x^3 + ax + b$$

并要求 $4a^3+27b^2 \neq 0$，$E: y^2=x^3+ax+b$ 可以构成群。

这个要求的原因是：$\Delta=(a/3)^3+(b/2)^2=(4a^3+27b^2)/108$ 是方程 $E: y^2=x^3+ax+b$ 的判别式，当 $4a^3+27b^2=0$，则方程有重根，设为 x_0，则点 $Q_0=(x_0,0)$是方程 $y^2=x^3+ax+b$ 的重根。令 $F(x,y)=y^2-x^3-ax-b$，则$\left.\frac{\partial F}{\partial x}\right|_{Q_0}=\left.\frac{\partial F}{\partial y}\right|_{Q_0}=0$，所以$\frac{\mathrm{d}y}{\mathrm{d}x}=-\frac{\partial F}{\partial x}\Big/\frac{\partial F}{\partial y}$在$Q_0$点无定义，即曲线 $E: y^2=x^3+ax+b$ 在 Q_0 点的切线无定义，因此在计算 Q_0 点的标量乘法运算时无定义。

2. 有限域 GF(p)上的椭圆曲线

密码学中普遍采用的是有限域上的椭圆曲线，就是指椭圆曲线方程定义式中，所有的系数都是某一有限域中的元素。这种有限域可以是 GF(q)，q 通常为一个素数幂，如最常见的情形是 $q=p$，p 为一个大于 2^{160} 的素数，或者 $q=2^m$($m>160$)。(本章只介绍第一种

情况，第二种情况可以类推进行定义。）

考虑有限域 GF(p)上的椭圆曲线。常见表达式为 $y^2 \equiv x^3+ax+b \bmod p$，其中 p 为一个大素数，a,b,x,y 均在有限域 GF(p)中，即从$\{0,1,\cdots,p-1\}$上取值，且满足：$4a^2+27b^2 \bmod p \neq 0$，这类椭圆曲线通常用 $E_p(a,b)$表示，该椭圆曲线只有有限个点数 N(包括无穷远点 O)，N 越大，安全性越高(群中元素越多，越能对抗穷举搜索攻击)。于是，一个很自然的问题就是，N 是否足够大。关于 N 的数量有一个估计，即 Hasse 定理。

定理 9.1　Hasse 定理。如果 E 是有限域 GF(p)上的椭圆曲线，N 是 E 上的点(x,y)(其中 $x,y\in Z_p$)的个数，则$|N-(p+1)|\leqslant 2\sqrt{p}$。即，$p+1-2\sqrt{p}\leqslant|N|\leqslant p+1+2\sqrt{p}$。

例如，若 $p=5$，Z_5 上的椭圆曲线 $y^2=x^3+ax+b$ 上的点数在 2～10 之间。

9.2　椭圆曲线群的构造

1. 椭圆曲线群的构造

一般来说，$E_p(a,b)$由以下方式产生：

(1)对每一 x ($0\leqslant x<p$，p 为整数)，计算 $t=x^3+ax+b \bmod p$。

(2)判定 t 在模 p 下是否为二次剩余(利用 Euler 准则)，如果不是，则曲线上没有与这一 x 相对应的点。如果有，则求出两个平方根($t=0$ 时只有一个平方根)。

例 9.1　$p=23$，$a=b=1$，方程为 $y^2=x^3+x+1$，$4a^3+27b^2=8\neq 0$，图形是连续曲线，如图 9.1 左图所示。$p=11$，$a=-1$，$b=0$，方程为 $y^2=x^3-x$，$4a^3+27b^2=-4\neq 0$，如图 9.1右图所示。

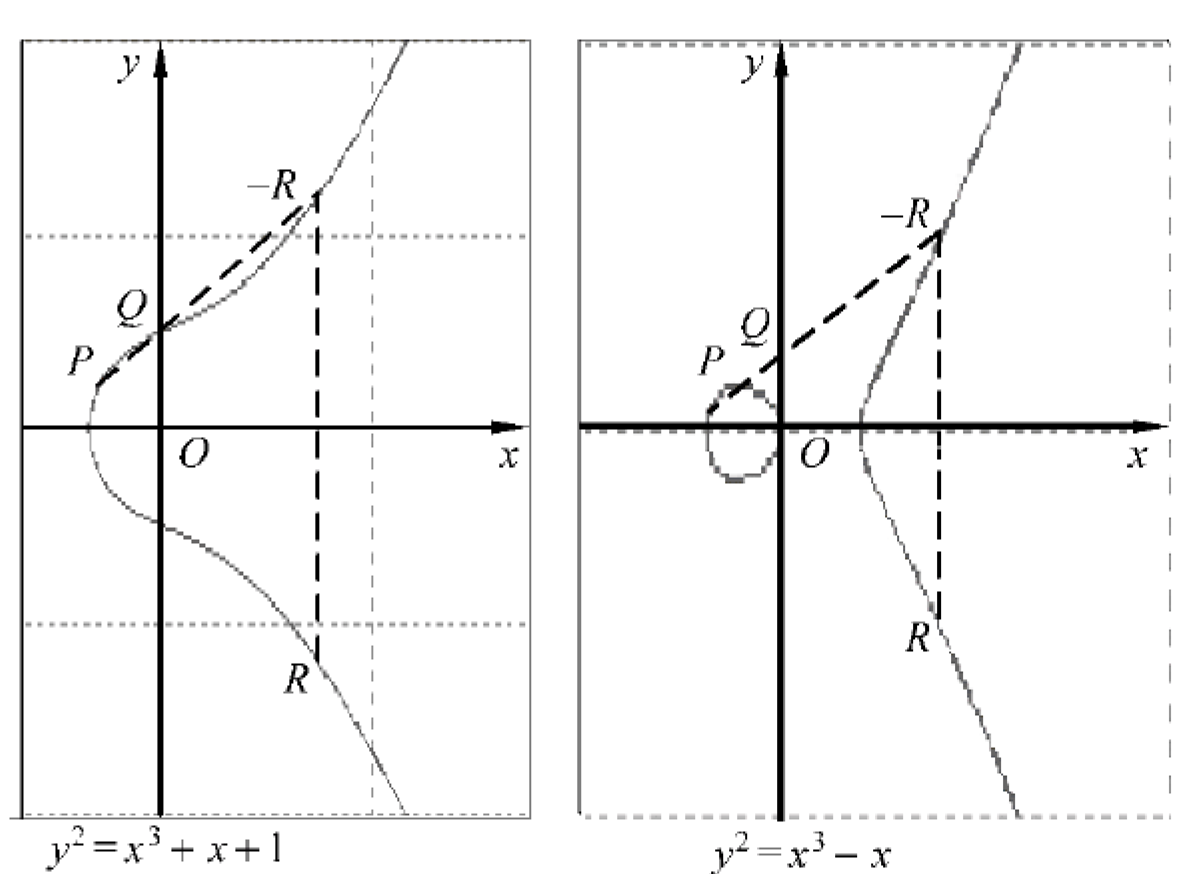


图 9.1　椭圆曲线 $y^2=x^3+x+1$ 和 $y^2=x^3-x$

主要对第一象限的整数点感兴趣，用 $E_p(a,b)$表示该方程定义的点集$\{(x,)|0\leqslant x<p,0\leqslant y<p,x,y\in Z\}\cup O$。表 9.1 给出了点集 $E_{23}(1,1)$。

图 9.2 给出了这些点的位置。

思考 9.1：给出点集 $E_{11}(-1,0)$。见表 9.2。

表 9.1 点集 $E_{23}(1,1)$

(0,1)	(1,7)	(3,10)	(4,0)	(5,4)	(6,4)	(7,11)
(0,22)	(1,16)	(3,13)		(5,19)	(6,19)	(7,12)
(9,7)	(11,3)	(12,4)	(13,7)	(17,3)	(18,3)	(19,5)
(9,16)	(11,20)	(12,19)	(13,16)	(17,20)	(18,20)	(19,18)

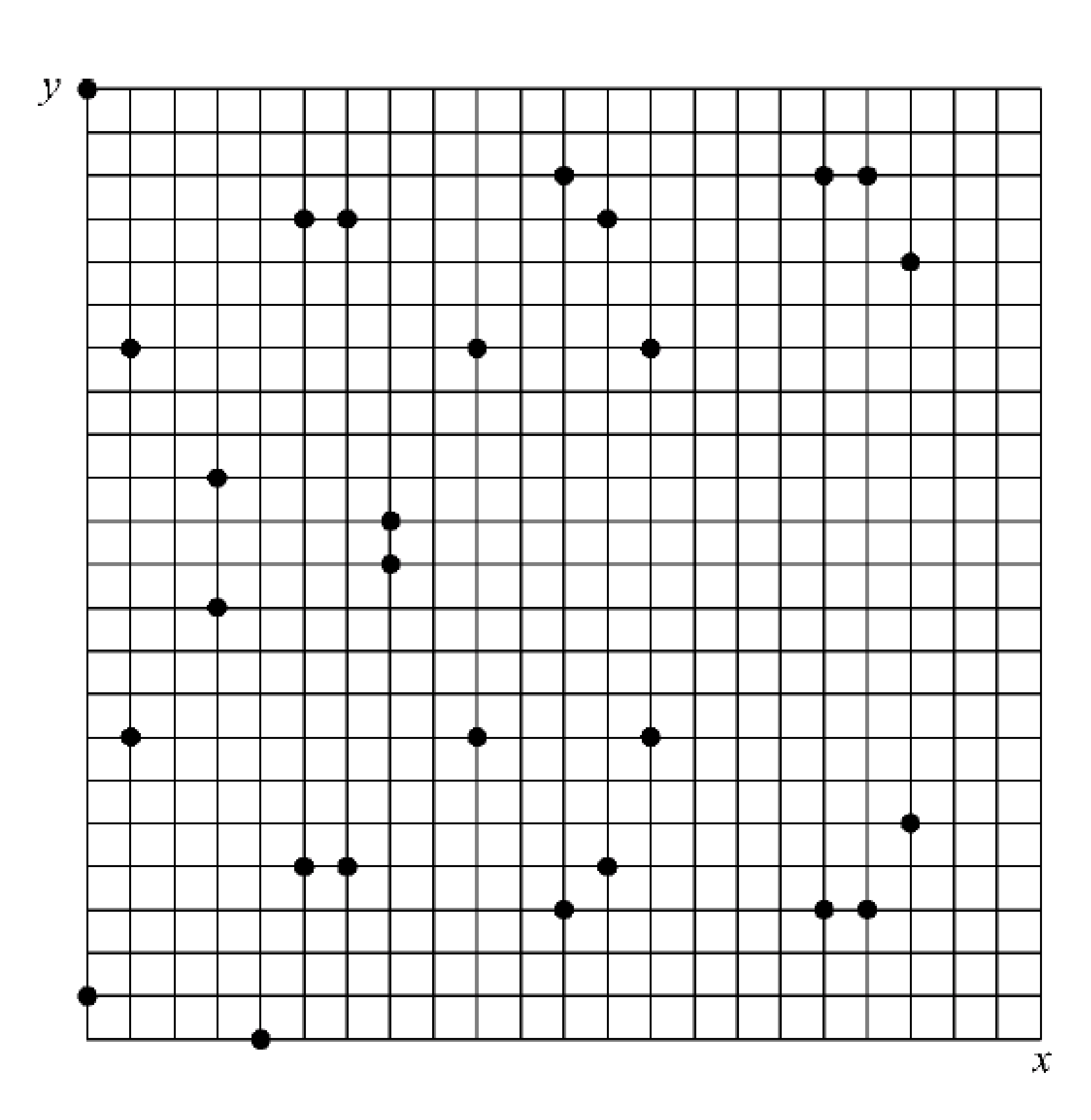


图 9.2 点集 $E_{23}(1,1)$

表 9.2 点集 $E_{11}(-1,0)$

(1,0)	(4,4)	(6,1)	(8,3)	(9,4)	(10,1)
(0,0)	(4,7)	(6,10)	(8,8)	(9,7)	(10,10)

2 椭圆曲线在 GF(p)下的 Abel 群

定理 9.2 椭圆曲线上的点集合 $E_p(a,b)$ 对于如下定义的加法规则构成一个 Abel 群。

椭圆曲线上的加法运算定义如下：如果其上的 3 个点位于同一直线上，那么它们的和为 O。

从图形上直观地进行定义和解释如下（见图 9.3）：

（1）O 是加法单位元，即对于椭圆曲线上任一点 P，其与关于 x 轴的对称点的连线与无穷远点 O 共线。$P+(-P)=O$，有 $P+O=P$；

（2）一条与 X 轴垂直的线和曲线相交于两个点，这两个点的 x 坐标相同，即 $P_1=(x,y)$ 和 $P_2(x,-y)$。P_1P_2 连线延长到无穷远时，与曲线相交于无穷远点 O，因此 3 点 P_1，P_2，O 共线，所以 $P_1+P_2+O=O$，$P_1+P_2=O$，即 $P_2=-P_1$，故椭圆曲线的性质决定 P 与其逆元成对在椭圆曲线上。

(3) 横坐标不同的两个点 P 和 Q 相加时，$P+Q$ 的定义如下：先在它们之间画一条直线并求直线与曲线的第三个交点 R，由 $P+Q+R=O$，得到 $P+Q=-R$；

(4) 两个相同点 P 相加时，通过该点画一条切线，切线与曲线交于另一点 R，则 $P+P+R=2P+R=O$，于是 $2P=-R$。类似可以定义 $3Q=Q+Q+Q$ 等。

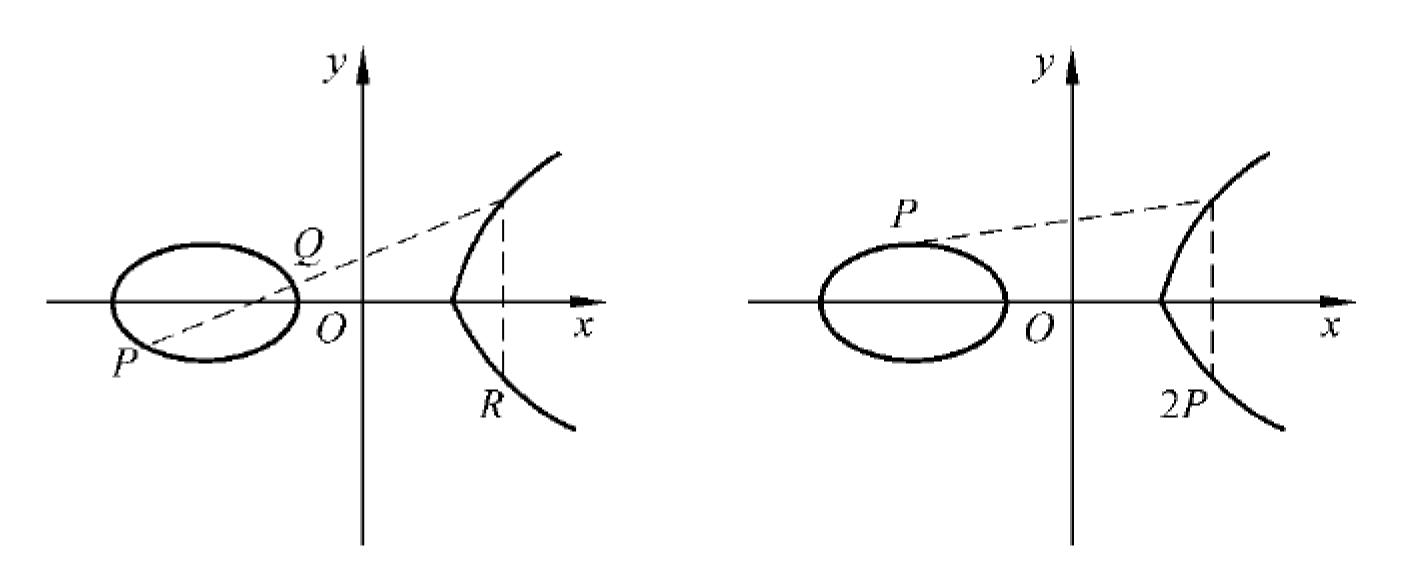


图 9.3　$P+Q=R$ 的几何意义(左图)和 $2P$ 的几何意义(右图)

可以证明，如上方法定义的加法运算可以得到一个 Abel 群。即可以得到如下加法规则：

(1) $O+O=O$；

(2) 对所有的点 $P(x,y)\in E_p(a,b)$，有 $P+O=O+P=P$；

(3) 对所有的点 $P(x,y)\in E_p(a,b)$，有 $P+(-P)=O$；即点 P 的逆为 $-P=(x,-y)$；

(4) 令 $P=(x_1,y_1)\in E_p(a,b)$，和 $Q=(x_2,y_2)\in E_p(a,b)$，且 $P\neq -Q$，则：

$$P+Q=R=(x_3,y_3)\in E_p(a,b)。$$

其中：$x_3=\lambda^2-x_1-x_2, y_3=\lambda(x_1-x_3)-y_1$

$$\lambda=\begin{cases}\dfrac{y_2-y_1}{x_2-x_1}, & P\neq Q\\[2ex]\dfrac{3x_2^2+a}{2y_1}, & P=Q\end{cases}$$

(5) 对于所有的点 P 和 Q，满足加法交换律，即 $P+Q=Q+P$；

(6) 对于所有的点 P、Q 和 R，满足加法结合律，即 $(P+Q)+R=P+(Q+R)$。

由规则(4)知，加法具有封闭性。规则(2)说明存在单位元 O。规则(3)说明任意元素 P 均存在逆元 $-P$。再加上规则(5)加法满足交换律和(6)加法满足结合律，故 $E_p(a,b)$ 对于椭圆曲线加法构成一个 Abel 群。规则(4)的计算在 GF(p)下进行，即对 p 取模。

例 9.2　椭圆曲线 $E_{23}(1,1)$，设 $P=(3,10)$，$Q=(9,7)$。求 $P+Q$，$2P$。

$$\lambda=(7-10)/(9-3)=-3/6\equiv 11 \bmod 23$$

$$x_3=11^2-3-9=109\equiv 17 \bmod 23$$

$$y_3=11(3-17)-10=-164\equiv 20 \bmod 23$$

于是，$P+Q=(17,20)$，可见其仍为 $E_{23}(1,1)$ 中的点。

下面求 $2P$。

$$\lambda=(3\cdot 3^2+1)/(2\cdot 10)=5/20=1/4\equiv 6 \bmod 23$$
$$x_3=6^2-3-3=30\equiv 7 \bmod 23$$
$$y_3=6(3-7)-10=-34\equiv 12 \bmod 23$$

所以 $2P=(7,12)$。

乘法规则：

(1) 如果 k 为整数，则对所有的点 $P\in E_p(a,b)$ 而言，有

$$kP=P+P+\cdots+P, k \text{ 个 } P \text{ 相加。}$$

(2) 如果 s 和 t 为整数，则对所有的点 $P\in E_p(a,b)$ 而言，有

$$(s+t)P=sP+tP, s(tP)=(st)P$$

9.3 椭圆曲线密码

9.3.1 椭圆曲线上的 DH 密钥协商协议

一个直观的类比是：定义在椭圆曲线群 $(E,+)$ 上的加法操作对应于 Z_p^* 上的模 p 乘法操作，多次加法操作对应于模 p 的指数运算。

1. 椭圆曲线离散对数问题(ECDLP)

设 p 某个大素数，E 是 $GF(p)$ 上的椭圆曲线，设 G 是 E 的一个循环子群，P 是 G 的一个生成元，$Q\in G$。已知 P 和 Q，求满足 $nP=Q$ 的唯一整数 n，$0\leqslant n\leqslant \mathrm{ord}(P)-1$，称为椭圆曲线离散对数问题。

ECDLP 其实就是 DLP 的椭圆曲线版本，即将原来的 Z_p^* 群替换成椭圆曲线上 Abel 群。

2. 椭圆曲线 Diffie-Hellman 问题(ECDHP)

同 ECDLP 一样，ECDHP 问题就是在 DHP 的椭圆曲线版本。

于是可以得到类似于 5.3 节介绍的 DH 密钥交换协议的椭圆曲线版本。描述如下：

(1) 假设 Alice 与 Bob 要在他们之间建立一个共享的密钥。Alice 和 Bob 首先选定公共参数：取某个大素数 p，E 是 $GF(p)$ 上的椭圆曲线，E_p 是相应的 Abel 群，G 是 E_p 中具有较大素数阶 n 的点。

(2) Alice 秘密选定一个整数 a：$2\leqslant a\leqslant n-1$，并计算 $A=aG$。发送 A 给 Bob。

(3) Bob 秘密选定一个整数 b：$2\leqslant b\leqslant n-1$，并计算 $B=bG$。发送 B 给 Alice。

(4) Alice 计算 $k=aB$。

(5) Bob 计算 $k=bA$。

容易看到，Alice 和 Bob 计算得到的 k 是相同的：

$$aB=a(bG)=(ab)G=b(aG)=bA$$

显然，椭圆曲线上 Diffie-Hellman 密钥交换协议的安全性基于 ECDLP 的困难性。

9.3.2 ElGamal加密的椭圆曲线版本

一个直接构造椭圆曲线上的公钥密码体制的方法是使用某种编码的方法，将明文编码为椭圆曲线上的一个点，然后利用 ElGamal 的思路，利用 DHP 的困难性，构造一个共享密钥来加密明文(某个椭圆曲线上的点)。

类比 ElGamal 密码体制(运算从模乘变为椭圆曲线群的加)，容易给出一种密码体制如下。

(1) **密钥生成**：设 E_p 是有限域 GF(p)上的椭圆曲线，G 是 E_p 中具有较大素数阶 n 的一个点。随机选择一个整数 d，使得 $2\leqslant d\leqslant n-1$，计算 $P=dG$。d 是私钥，(P,G,E,n)是公钥。

(2) **加密算法**：将明文编码为 E_p 中的元素 P_m(即椭圆曲线上的一个点)，再选取随机数 r：$1\leqslant r\leqslant n-1$，计算

$$c_1=r\cdot G=(x_1,y_1)$$

$$c_2=P_m+r\cdot P=(x_2,y_2)$$

(3) **解密算法**：利用私钥 d，计算出 $P_m=c_2-d\cdot c_1$。对 P_m 解码得到明文 m。

例 9.3 取 $p=751$，$E_p(-1,188)$，即椭圆曲线为 $y^2=x^3-x+188$，$E_{751}(-1,188)$的一个生成元是 $G=(0,376)$，A 的公钥为 $P=(201,5)$。假定 B 已将要发送给 A 的消息嵌入到椭圆曲线上，即点 $P_m=(562,201)$，B 选取随机数 $r=386$，由 $rG=386(0,376)=(676,558)$，$P_m+rP=(562,201)+386(201,5)=(385,328)$。得到的密文为{(676,558)，(385,328)}。

9.3.3 椭圆曲线快速标量点乘算法

从密码学应用可以看到，在椭圆曲线群中最重要的运算是标量乘，即对任意正整数 m 及 $E_p(a,b)$中非零元 P，计算 $m\cdot P$。计算标量乘的方法有很多种，最常用的方法是倍加法，即将乘子 m 表示成系数为 0 或 1 的 2 的多项式，然后反复进行“倍－加”及不同点加的运算。

例 9.4 $E_{17}(-1,5)$中 $P=(7,1)$，求 mP。

$$\begin{aligned}mP &= 15(7,1) = 2(2(2(7,1)+(7,1))+(7,1))+(7,1)\\ &= 2(2(11,13)+(7,1))+(7,1))+(7,1)\\ &= 2(2(8,13)+(7,1))+(7,1)\\ &= 2((14,7)+(7,1))+(7,1)\\ &= 2(12,2)+(7,1)\\ &= (8,4)+(7,1)\\ &= (11,4)\end{aligned}$$

计算 15·(7,1)需要进行 3 次“倍－加”和 3 次加运算。如果采用符合二元法(符合二元法是指：将乘子表示成系数为 0、1 或－1 的 2 的多项式，然后再反复进行“倍－加”及

不同点加的一种标量乘方法)，则只需要 4 次“倍—加”与 1 次加，即：

$$
\begin{aligned}
mP &= 15(7,1) = (2^4 - 1)(7,1) \\
&= 2(2(2(2(7,1)))) - (7,1)) \\
&= 2(2(2(11,13))) - (7,1) \\
&= 2(2(10,14)) - (7,1) \\
&= 2(13,8) - (7,1) \\
&= (7,16) - (7,1) \\
&= (7,16) + (7,16) \\
&= (11,4)
\end{aligned}
$$

思考 9.2：这一算法和 RSA 中用到的(见 3.3.3 节)计算模幂的“平方—乘”(模重复平方法)方法是否构成类比。

其实，类比关系十分明显。在模幂运算中，“乘”为基本运算，椭圆曲线标量乘中“加”为基本运算。“平方”为两次“乘”，“倍”为两次“加”。因此构成非常明显的类比。

思　考　题

[1] 编写程序实现椭圆曲线快速标量点乘算法。

[2] 编写程序实现椭圆曲线版本的 ElGamal 加密。

第 10 章 大整数分解算法

对 RSA 最直接的攻击就是因子分解。如果能够分解 n 得到 p 和 q，便可以得到 $\phi(n)=(p-1)(q-1)$。根据公钥 e 求得私钥 $d\equiv e^{-1} \bmod \phi(n)$，便完全攻破。因子分解的平凡办法就是试除法，其基本思想是尝试小于$\sqrt{n}$的所有素数去除 n，直到找到一个因子，这种方法当 n 较大时，在现实中是不可行的。

本章的重点是随机平方法，难点是 Pollard p－1 分解算法。

10.1 Pollard Rho 方法

Rho 方法由 Pollard 于 1975 年提出，用于寻找小因子。基于一种在有限集合中寻找碰撞的思想，其基本原理是：设 $f:s\rightarrow s$ 是一个随机函数，S 是 n 个元素的有限集。设 x_0 是 S 中的一个随机元素，考查序列 x_0, x_1，其中 $x_{i+1}=f(x_i)$，$i\geqslant 0$。由于 S 是有限集，则该序列最终必然出现循环。

例 10.1 函数 f：$\{1,2,\cdots,13\}\rightarrow\{1,2,\cdots,13\}$。定义如下：

$f(1)=4, f(2)=11, f(3)=1, f(4)=6, f(5)=3, f(6)=9, f(7)=3,$

$f(8)=11, f(9)=1, f(10)=2, f(11)=10, f(12)=4, f(13)=7$

容易画出函数图如图 10.1 所示。

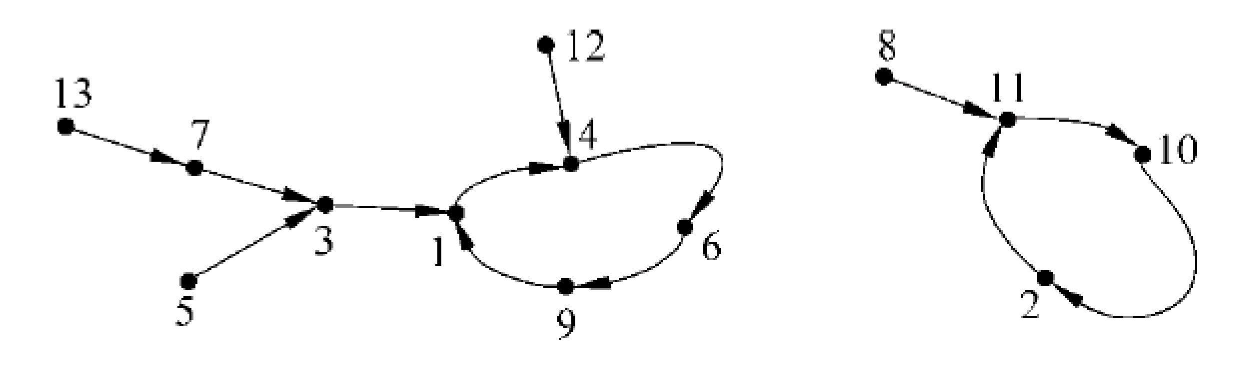


图 10.1 随机映射 f 的函数图

很明显一个问题是：大概多少步之后出现循环。即出现碰撞需要的步数。

设走到循环开始的位置需要 u 步(也称为尾长度)，循环的长度为 v 步(称为圈长)，因此，发现碰撞需要的步数为 $u+v$(也称为 ρ 长度，ρ 字母的形状就是由尾和圈组成的，ρ 在希腊字母中叫做 Rho)。关于随机映射或者随机图的研究表明，u 的期望是 $\sqrt{\pi n/8}$，圈长度的期望也是 $\sqrt{\pi n/8}$，于是 ρ 长度为 $\sqrt{\pi n/2}$。Rho 方法在解决离散对数问题中也有应用。

下面给出 Pollard 的 Rho 因子分解算法。

算法 10.1　Rho(n)。

```
/* Pollard的 rho因子分解算法           */
/* 输入：合数 n,n不为某个素数的幂       */
/* 输出：n的非平凡因子 d               */
{
  a←2,b←2
  For i=1 To ...
  {
  a←a²+1 mod n;
  b←b²+1 mod n;b←b²+1 mod n;
  d=gcd(a-b,n);
  If 1<d<n, Return(d);
  If d=n, Return("Failure");
}
```

这里的算法在平凡的 rho 方法的基础上作了一点改进，即计算(x_i, x_{2i})(a 计算一次，b 计算两次)，该方法称为 Floyd 循环查找算法，可节省存储的空间。

10.2　Pollard p－1 分解算法

这种方法的思想是：设 p 是要分解的数 n 的一个素数因子，$p-1$ 的因子分解中的素数幂次最大为 q，$q|p-1$，如果 $q\leqslant B$，那么必有 $(p-1)|B!$。例如，假设 $p-1=5280=2^5\cdot 3\cdot 5\cdot 11$，素数的幂次最大为 $2^5=32$，可选择 $B=32$。

由 Fermat 定理，$2^{p-1}\equiv 1 \bmod p$，于是 $2^{B!}\equiv 1 \bmod p$。即 $p|(2^{B!}-1)$，又因为 $p|n$，所以 $p|\gcd(n, 2^{B!}-1)$。于是可以得到一个 n 的非平凡因子 $\gcd(n, 2^{B!}-1)$。

下面给出 Pollard p－1 算法，算法中的 B 是一个猜测值，即对 $p-1$ 中最大素数幂的猜测。

算法 10.2　PollardFactoring(n,B)。

```
/*  分解 n的因子 */
/* 输入：要分解的合数 n,对 p-1的因子分解中的素数幂的最大值的猜测 B*/
{
    a←2;
    For j←2 To B
    {  a←a^j mod n;
    }
    d←gcd(a-1,n);
    If 1<d<n Return(d);
```

```
    Else
    Return ("Failure");
}
```

另外一种方法也可以得到 $p-1$ 的倍数，可能会缩小计算量，这里先引入光滑(smooth)的概念，简单地说，就是素数因子的上界。

定义 10.1 设 B 是一个正整数。当 n 的所有素数因子不大于 B 时，称整数 n 为 B 光滑的(B Smooth)。

例如，假设 $p-1=5280=2^5\cdot 3\cdot 5\cdot 11$，则 $p-1$ 是 11 光滑的($B=11$)。

下面考虑如何算出一个数 Q，满足 $(p-1)|Q$ 且计算量较小。一个自然的想法是假设 $p-1$ 是 B 光滑的，那么计算 $Q=\prod_{q\leqslant B_q}$，即小于 B 的素数的乘积，但是，该数可能不满足 $(p-1)|Q$，因为没有考虑 $p-1$ 中素因子可能有幂次。于是，要适当放大 Q。考查某个 $p-1$的因子 q，其最大的幂次为 l，$q^l\leqslant n$，于是 $l\leqslant\left\lfloor\frac{\ln n}{\ln q}\right\rfloor$。这样，令

$$Q=\prod_{q\leqslant B}q^{\left\lfloor\frac{\ln n}{\ln q}\right\rfloor}$$

这样，必然有 $(p-1)|Q$，于是 $p|2^Q-1)$，$\gcd(2^Q-1,n)$即为 n 的非平凡因子。下面给出算法。

算法 10.3 PollardFactoring(n,B)。

```
/* 分解 n的因子 */
/* 输入：要分解的合数 n,对 p-1的素因子的最大值的猜测 B*/
{
    a←2,q←2;
    While (q≤B and q∈Prime)
    {l=⌊lnn/lnq⌋;
    a←a^(q^l) mod n;
    }
    d←gcd(a-1,n)
    If 1<d<n Return(d);
    Else
    Return("Failure");
}
```

10.3 随机平方法

该方法的基本思想是：假如 x 和 y 是整数，且 $x^2\equiv y^2 \bmod n$，但 $x\neq\pm y \bmod n$。于是 n 整除 $x^2-y^2=(x+y)(x-y)$，但是 n 不能整除$(x-y)$或者$(x+y)$。因此 $\gcd(x+$

y,n)(或者说 $\gcd(x-y,n)$)一定是 n 的一个平凡因子。例如，$10^2 \equiv 32^2 \bmod 77$，则 $\gcd(10+32,77)=7$ 是 77 的一个因子。

通常的策略是随机找几个数，不妨称为 z_i，这些数的平方模 n 的素数因子都在一个集合内，这个集合称为因子基 B。如果把这些 z_i^2 相乘，发现其相同素数因子的幂次为偶数次，即意味着是一个数 t 的平方。于是找到了两个数$(\prod(z_i))^2=t^2$。

例 10.2　假定 $n=15770708441$，令 $B=2,3,5,7,11,13$。设法找到几个 z_i，使得其平方后取模的因子在这个集合中。考虑如下：

$$z_1=8340935156, z_2=12044942944, z_3=2773700011$$

$$z_1^2=8340935156^2=3 \cdot 7 \bmod n$$

$$z_2^2=12044942944^2=2 \cdot 7 \cdot 13 \bmod n$$

$$z_3^2=2773700011^2=2 \cdot 3 \cdot 13 \bmod n$$

把它们相乘，得到$(z_1 \cdot z_2 \cdot z_3)^2=(2 \cdot 3 \cdot 7 \cdot 13)^2 \bmod n$。即 $9503435785^2=546^2 \bmod n$。然后计算 $\gcd(9503435785+546,n)$得到 n 的素数因子。

但是，问题是如何得到这些 z_i，使得它们的素数因子能在集合 B 中。通常集合 B 中的素数因子都不大，这意味着 z_i^2 模 n 比较小，即 z_i^2 和 n 的倍数比较接近。于是一个自然的想法是，令 $z_i=j+\lceil\sqrt{kn}\rceil$，$j=0,1,2,\cdots$，$k=1,2,\cdots$。这些数平方后一般是模 n 后大于 0 的。如果取 $z_i=$?kn ?，则通常平方模 n 会比 n 小一点。因此，也把 -1 也加入到 B 集合中。

例 10.3　假设 $n=1829$，取 $B=-1,2,3,5,7,11,13$。计算$\sqrt{n}=42.8$，$\sqrt{2n}=60.5$，$\sqrt{3n}=74.1$，$\sqrt{4n}=85.5$。假定取 $z=42,43,60,61,74,75,85,86$。可得到 $z^2 \bmod n$ 在 B 上的分解。

$$z_1^2 \equiv 42^2 \equiv -65 \equiv (-1) \cdot 5 \cdot 13$$

$$z_2^2 \equiv 43^2 \equiv 20 \equiv 2^2 \cdot 5$$

$$z_3^2 \equiv 61^2 \equiv 63 \equiv 3^2 \cdot 7$$

$$z_4^2 \equiv 74^2 \equiv -11 \equiv (-1) \cdot 11$$

$$z_5^2 \equiv 85^2 \equiv -91 \equiv (-1) \cdot 7 \cdot 13$$

$$z_6^2 \equiv 86^2 \equiv 80 \equiv 2^4 \cdot 5$$

如果将这些分解结果用 B 中元素的奇数或者偶次幂的向量表示，则更加清晰。例如，

$$z_1=(1,0,0,1,0,0,1), z_2=(0,0,0,1,0,0,0), z_3=(0,0,0,0,1,0,0)$$

$$z_4=(1,0,0,0,0,1,0), z_5=(1,0,0,0,1,0,1), z_6=(0,0,0,1,0,0,0)$$

容易观察到，$z_1 \cdot z_2 \cdot z_3 \cdot z_5$ 能够使 B 中元素出现的幂次为偶数，即找到一个平方数 $(42 \cdot 43 \cdot 61 \cdot 85)^2 \equiv (2 \cdot 3 \cdot 5 \cdot 7 \cdot 13)^2 \equiv 901^2 \bmod 1829$。于是 $\gcd(1459+901,1829)=59$ 得到一个非平凡因子 59。

前面介绍了 3 种先驱算法。实际中大整数因子分解最有效的算法是：Lenstra 椭圆

曲线因子分解法(Pollard p－1 方法用于随机椭圆曲线群)、Pomerance 的二次筛法(Quadratic Sieve)、Pollard 数域筛法(Number Field Sieve)(两者都来源于随机平方法)。最近发展起来的是数域筛法，其渐进时间比其他两个算法都要短。

思 考 题

编写程序实现 3 种大数分解算法。

第 11 章 离散对数算法

上一章介绍了公钥密码学中一个重要的困难问题——大数分解问题的算法，本章介绍公钥密码学中另一个困难问题——离散对数的算法。

本章的重点是指数演算法，难点是 Pohlig-Hellman 算法。

11.1 小步大步算法

先来看穷举搜索法。这是一种平凡方法，对任意群有效，即计算 $\alpha^0,\alpha^1,\alpha^2,\cdots$，直到得到 β 为止。该方法要 $O(n)$ 次乘法，这里 n 是 α 的阶，且 n 较大时，该算法的效率很低。

小步大步算法(Baby-Step Giant Step)

该算法由 Shank 提出，故也叫做 Shank 算法。该算法是通用算法，对任意群有效。假定需要求的是在群 G 中的 $x=\log_\alpha\beta$。

如果令 $m=\lceil\sqrt{n}\,\rceil$，n 为群的阶，则 x 表示为 $x=mj+i$，$(0\leqslant i,j\leqslant m-1)$，因此 $\beta=\alpha^x=\alpha^{mj}\alpha^i$，即 $\beta(\alpha^{-i})=\alpha^{mj}$。

思考 11.1 如何让等式 $\beta(\alpha^{-i})=\alpha^{mj}$ 成立。

等式中除了 i,j 未知以外，其他量均已知，所以目标是找到 i,j。由于 $0\leqslant i,j\leqslant m-1$，且计算 α^{mj} 的计算量较小，于是可以先计算出 m 个值：

$$1,\alpha^m,(\alpha^m)^2,(\alpha^m)^3,\cdots,(\alpha^m)^{m-1}$$

不妨称这些值为一个查询表，用 L 表示。然后去找 i，办法是计算 $\beta,\beta\alpha^{-1},\beta\alpha^{-2},\cdots,\beta\alpha^{-i},\cdots$。注意，每计算一个数，就和查询表对照，看是否有相等的。如果发现有相等的，即找到 i,j。求出 $x=mj+i$。

一个常见的优化方法是构造有序的查询表，这样在查找的时候更快一些。查询表的构造需要 $O(n)$ 时间计算 α 的 n 个幂，$O(n\log n)$ 时间对 n 个元素排序(如快速排序)。如果忽略 $O(\log n)$，则预先计算的时间为 $O(n)$。预先计算不考虑在计算离散对数问题所耗时间，于是小步大步算法可以 $O(1)$ 时间内完成(对 n 个有序元素的查找需要 $O(\log n)$ 时间(如二分查找法，不考虑)，需要的存储空间为 $O(n)$)。这结果比穷举搜索法的计算时间 $O(n)$ 要好，但是穷举搜索需要的空间为 $O(1)$。可见，小步大步算法是用空间代价换取了时间效率。

算法 11.1 Baby-Step-Giant-Step(G,n,α,β)。

```
/* Baby Step Giant Step 算法,计算 logαβ */
/* 输入:任意循环群 G,阶为 n,α,β */
/* 输出:logαβ */
{
m←⌈√n⌉
For j←0 To m-1
computing α^mj
save and sort (j,α^mj) (call it L)
For i←0 To m-1
y←β(α^-i)
Search y in L If y=α^mj Return (mj+i)
}
```

例 11.1 设在群 Z_{809}^* 中计算 $\log_3 525$。

809 是素数,3 是 Z_{809}^* 的生成元,该群的阶为 808。有 $\alpha=3,\beta=525,m=\lceil\sqrt{n}\rceil=\lceil\sqrt{808}\rceil=29,\alpha^m \bmod 809=3^{29} \bmod 809=99$。

$$(j,99^j \bmod 809)=[(1,0),(1,99),(2,93),(3,308),(4,559),(5,329),(6,211),(7,664),(8,207),(9,268),(10,644),(11,654),(12,26),(13,147),(14,800),(15,727),(16,781),(17,464),(18,632),(19,275),(20,528),(21,496),(22,564),(23,15),(24,676),(25,586),(26,575),(27,295),(28,81)]$$

计算

$$(i,\beta(\alpha^{-i}))=(i,525\cdot(3^i)^{-1})=[(0,525),(1,175),(2,328),(3,379),(4,396),(5,132),(6,44),(7,544),(8,724),(9,511),(10,440),(11,686),(12,768),(13,256),(14,355),(15,388),(16,399),(17,133),(18,314),(19,644)]$$

可以发现,$j=10,i=19$ 时,出现相等。于是 $\log_3 525=(29\cdot 10+19) \bmod 809=309$。

11.2 Pollard Rho 算法

Rho 方法是可适用于任意群的通用方法。其基本思想是:通过迭代计算一个随机函数 f,构造一个序列 $x_1,x_2,\cdots$。一旦在序列中得到两个元素 x_i,x_j,满足 $x_i=x_j,(i<j)$,就有希望计算出 $\log_\alpha\beta$。为了节省空间,通常寻找碰撞 $x_i=x_{2i}$。

思考 11.2 什么迭代函数能够在找到碰撞后求解 $\log_\alpha\beta$。

考虑群 Z_p^*,p 为素数。形如 $x_i=\alpha^{a_i}\beta^{b_i} \bmod p$ 的随机函数(构成一个随机图)。如果找到碰撞 $x_{2i}=x_i(i\geqslant 1)$,则 $\alpha^{a_i}\beta^{b_i}=\alpha^{a_{2i}}\beta^{b_{2i}} \bmod p$,于是

$$\alpha^{a_i - a_{2i}} = \beta^{b_{2i} - b_i} \bmod p$$
$$\alpha^{\frac{(a_i - a_{2i})}{(b_{2i} - b_i)}} = \beta \bmod p$$
$$\log_\alpha \beta = (b_{2i} - b_i)^{-1}(a_i - a_{2i}) \bmod n$$

这里 n 是 α 的阶。一般来说 $b_{2i} - b_i = 0$ 的概率可以忽略。先将群中元素均匀分成 3 个集合，x 在不同集合中的迭代方程不同，便于更快地找到碰撞。令

$$x_{i+1} = f(x_i) = \begin{cases} \beta \cdot x_i, & x_i \in S_1 \\ x_i^2, & x_i \in S_2 \\ \alpha \cdot x_i, & x_i \in S_3 \end{cases}$$

为了在迭代中保持 $x_i = \alpha^{a_i}\beta^{b_i}$ 不变，显然 $x_i \in S_1$ 时，$\alpha^{a_{i+1}}\beta^{b_{i+1}} = \beta x_i = \beta\alpha^{a_i}\beta^{b_i} = \alpha^{a_i}\beta^{b_i+1}$，故有 $a_{i+1} = a_i, b_{i+1} = b_i + 1$。同理，得到关于 a_i, b_i 的递推关系：

$$a_{i+1} = \begin{cases} a_i, & x_i \in S_1 \\ 2a_i \bmod n, & x_i \in S_2 \\ a_i + 1 \bmod n, & x_i \in S_3 \end{cases}, \quad b_{i+1} = \begin{cases} b_i + 1 \bmod n, & x_i \in S_1 \\ 2b_i \bmod n, & x_i \in S_2 \\ b_i, & x_i \in S_3 \end{cases}$$

因此，如果将 (x, a, b) 视为一个三元组，则有

$$f(x, a, b) = \begin{cases} (\beta x, a, b+1), & x_i \in S_1 \\ (x^2, 2a, 2b), & x_i \in S_2 \\ (\alpha x, a+1, b), & x_i \in S_3 \end{cases}$$

于是可以得到算法如下：

算法 11.2：Rho-DLP(G，n，α，β)。

```
/* 计算离散对数的 Pollard Rho 算法 */
/* 输入：阶为素数 n 的循环群 G 的一个生成元 α，一个元素 β∈G */
/* 输出：离散对数 x=log_α β */
Procedure f(x,a,b)
{
If x∈S1    f←(β·x,a,(b+1) mod n);
else if x∈S2    f←(x^2,2a mod n, 2b mod n);
else
f←(α·x,(α+1) mod n,b);
Return (f);
}
main()
{
G= S1∪S2∪S3;
(x,a,b)←f(1,0,0);
(x′,a′,b′)←f(x,a,b);
While x≠x′ Do{
(x,a,b)←f(x,a,b);
(x′,a′,b′)←f(x′,a′,b′);
```

```
(x′,a′,b′)←f(x′,a′,b′);
If gcd(b′- b,n)≠1 Return ("Failure");
Else Return ((a- a′)(b- b′) mod n))};
}
```

例 11.2 整数 $p=809$ 是素数，$\alpha=89$，$\beta=618$，计算 $\log_\alpha\beta$。

将所有数分成三个集合：

$$S_1=\{x\in Z_{809}:\ x\equiv 1 \bmod 3\}$$
$$S_2=\{x\in Z_{809}:\ x\equiv 0 \bmod 3\}$$
$$S_3=\{x\in Z_{809}:\ x\equiv 2 \bmod 3\}$$

从(1,0,0)开始迭代，第一次计算运用迭代算式 1($x\in S_1$)，可以得到如下计算过程：

i	(x_i,a_i,b_i)	(x_{2i},a_{2i},b_{2i})	i	(x_i,a_i,b_i)	(x_{2i},a_{2i},b_{2i})
1	(618,0,1)	(76,0,2)	6	(488,1,5)	(683,7,11)
2	(76,0,2)	(113,0,4)	7	(555,2,5)	(451,8,12)
3	(46,0,3)	(488,1,5)	8	(605,4,10)	(344,9,13)
4	(113,0,4)	(605,4,10)	9	(451,5,10)	(112,11,13)
5	(349,1,4)	(422,5,11)	10	(422,5,11)	(422,11,15)

可见，发生碰撞是 x_{10}，$=x_{20}=422$，$\alpha=89$ 在 Z_{809}^* 的阶为 101。故

$$\log_\alpha\beta=(15-11)^{-1}(5-11) \bmod 101=6\cdot 4^{-1} \bmod 101=49$$

算法中如果 $\gcd(b'-b,n)>1$，则算法会停止并输出“Failure”，这种情况其实并不是完全不能得到答案，如果 $\gcd(b'-b,n)=d$，可证明同余方程 $c(b'-b)=a-a' \bmod n$，($c=\log_\alpha\beta$)有 d 个解。假如 d 不是很大，可以直接算出 d 个解进行验证。

最后看看该算法的效率。同前面因子分解问题的 Rho 算法一样，在 n 阶循环群中平均经过 $O(\sqrt{n})$ 次迭代(随机图上行走的圈与尾的长度和的期望值，即 ρ 的期望值)，会发生碰撞。

11.3 指数演算法

下面看一个对特殊的群有效的算法。假定 $B=\{p_1,p_2,\cdots,p_B\}$ 是一个“较小”素数的集合(因子基)，指数演算法(Index Calculus)的基本思想利用 $\log_\alpha p_i$，($1\leqslant i\leqslant B$)，来计算 $\log_\alpha\beta$。如何计算 $\log_\alpha p_i$，$1\leqslant i\leqslant B$，选取 $1\leqslant x\leqslant n-2$，使得 $a^x \bmod n$ 的素因子都在集合 B 中，考查 $\alpha^x\equiv p_1^{a_1}p_2^{a_2}\cdots p_B^{a_B} \bmod n$，即

$$x\equiv a_1\log_\alpha p_1+a_2\log_\alpha p_2+\cdots+a_B\log_\alpha p_B \bmod (n-1)$$

这个方程中有 B 个未知数 $\log_\alpha p_i(1\leqslant i\leqslant B)$，因此，随机选取 B 个 x_j，$1\leqslant x_j\leqslant n-2$，$1\leqslant j\leqslant B$，可以得到 B 个相互独立的同余方程：

$$x_j \equiv a_{1j}\log_\alpha p_1 + a_{2j}\log_\alpha p_2 + \cdots + a_{Bj}\log_\alpha p_B \bmod (n-1),\ 1 \leqslant j \leqslant B$$

由这些方程可以计算出 $\log_\alpha p_i, 1 \leqslant i \leqslant B$。

下面考虑如何计算 $\log_\alpha \beta$。随机选择 $s, 1 \leqslant s \leqslant n-2$，使得 $\beta\alpha^s \bmod n$ 的素数因子都在集合 B 中，于是

$$\beta\alpha^s \equiv p_1^{c_1} p_2^{c_2} \cdots p_B^{c_B} \bmod n$$

即

$$\log_\alpha \beta + s \equiv c_1 \log_\alpha p_1 + c_2 \log_\alpha p_2 + \cdots + c_B \log_\alpha p_B \bmod (n-1)$$

在该式中，除了 $\log_\alpha \beta$ 未知外，其他都是已知的。于是可以求出 $\log_\alpha \beta$。最后再次强调：B 个 $x_j, 1 \leqslant x_j \leqslant n-2, 1 \leqslant j \leqslant B$ 和 $s, 1 \leqslant s \leqslant n-2$，都是随机选择的，保留那些满足所有分解后的因子都在因子基 B 中的随机值，否则重试。

例 11.3 设群 Z_n^*，$n=10007$ 为素数，$\alpha=5$，$\beta=9451$，求 $\log_\alpha \beta$。

令 $B=\{2,3,5,7\}$。显然 $\log_5 5=1$，故只需要计算 $\log_5 2, \log_5 3, \log_5 7$。选取 $x=4063$，5136，9865，计算

$$5^{4063} \bmod 10007 = 42 = 2 \cdot 3 \cdot 7$$
$$5^{5136} \bmod 10007 = 54 = 2 \cdot 3^3$$
$$5^{9865} \bmod 10007 = 189 = 3^3 \cdot 7$$

于是有

$$\log_5 2 + \log_5 3 + \log_5 7 \equiv 4063 \bmod 10006$$
$$\log_5 2 + 3\log_5 3 \equiv 5136 \bmod 10006$$
$$3\log_5 3 + \log_5 7 \equiv 9865 \bmod 10006$$

可以求得 $\log_5 2=6578$，$\log_5 3=6190$，$\log_5 7=1301$。下面计算 $\log_5 9451$。选取 $s=7736$，计算 $9451 \cdot 5^{7736} \bmod 10007 = 8400 = 2^4 \cdot 3 \cdot 5^2 \cdot 7$。于是有

$$\begin{aligned}\log_5 9451 &= 4\log_5 2 + \log_5 3 + 2 + \log_5 7 - s \bmod 10006 \\ &= 4 \cdot 6578 + 6190 + 2 + 1301 - 7736 \bmod 10006 \\ &= 6057\end{aligned}$$

容易验证，$5^{6057} \bmod 10007 = 9451$。

11.4 Pohlig-Hellman 算法

该算法适用于群 Z_n^*，n 是一个素数，$n-1$ 的素数因子都是小素数的情况。设 $n-1=q_1^{e_1} q_2^{e_2} \cdots q_k^{e_k}$，其中 q_i 是素数，$1 \leqslant i \leqslant k$。目的是计算 $\log_\alpha \beta$，也就是寻找 $a, 0 \leqslant a \leqslant n-2$，使得 $\alpha^a = \beta \bmod n$。如果能求得

$$a \bmod q_i^{e_i}, \quad i=1,2,\cdots,k$$

则根据中国剩余定理，可以求得 $a \bmod (n-1)$，即求得 $\log_\alpha \beta$。

为了求 $a \bmod q_i^{e_i}$，$i=1,2,\cdots,k$，设

$$a \bmod q_i^{e_i} = a_0 + a_1 q_i + a_2 q_i^2 + \cdots + a_{e_i-1} q_i^{e_i-1}$$

其中，$0 \leqslant a_j < q_i$，$0 \leqslant j \leqslant e_i - 1$。下面的目标就是确定 $a_j (0 \leqslant j \leqslant e_i - 1)$。

因 $\beta \equiv \alpha^a \bmod n$，且由 Fermat 定理知，$\alpha^{n-1} \equiv 1 \bmod n$，故

$$\begin{aligned}\beta^{(n-1)/q_i} &\equiv \alpha^{a(n-1)/q_i} \bmod n \\ &\equiv \alpha^{(a_0+a_1q_i+a_2q_i^2+\cdots+a_{e_i-1}q_i^{e_i-1})(n-1)/q_i} \bmod n \\ &\equiv \alpha^{a_0(n-1)/q_i} \bmod n\end{aligned}$$

因此，为了确定 a_0，可以通过计算 $\beta^{(n-1)/q_i} \bmod n$，然后穷举 $\alpha^{s(n-1)/q_i} \bmod n(s=0,1,2,\cdots,q_i-1)$，如果发现两者相等，则此时的 $s=a_0$，从而确定了 a_0。

下面进一步确定 a_1。令 $\beta_1=\beta^{-a_0}$。因为

$$\begin{aligned}\beta_1^{(n-1)/q_i^2} &\equiv \alpha^{(a-a_0)(n-1)/q_i^2} \bmod n \\ &\equiv \alpha^{(a_1q_i+a_2q_i^2+\cdots+a_{e_i-1}q_i^{e_i-1})(n-1)/q_i^2} \bmod n \\ &\equiv \alpha^{a_1(n-1)/q_i} \bmod n\end{aligned}$$

同样的，穷举 $\alpha^{s(n-1)/q_i} \bmod n(s=0,1,2,3,\cdots,q_i-1)$，如果发现两者相等，则此时的 $s=a_1$。可见，穷举 $\alpha^{s(n-1)/q_i} \bmod n(s=0,1,23,\cdots,q_i-1)$ 只需要一次，将其保存为一个列表，不妨设为 L。一般地，(使用数学归纳法)假设已经求得 $a_0,a_1,\cdots,a_{j-1}$，$1\leqslant j\leqslant e_i-1$。令 $\beta_j=\beta\alpha^{-(a_0+a_1q_i+\cdots+a_{j-1}q_i^{j-1})}$，因为

$$\begin{aligned}\beta_j^{(n-1)/q_i^{j+1}} &\equiv \alpha^{(a-a_0-a_1q_i-\cdots-a_{j-1}q_i^{j-1})(n-1)/q_i^{j+1}} \bmod n \\ &\equiv \alpha^{(a_jq_i^j+a_{j+1}q_i^{j+1}+\cdots+a_{e_i-1}q_i^{e_i-1})(n-1)/q_i^{j+1}} \bmod n \\ &\equiv \alpha^{a_j(n-1)/q_i} \bmod n\end{aligned}$$

在表 L 中查询满足条件的 $s=a_j$。

如上所述，可以确定 $a_0,a_1,\cdots,a_{e_i-1}$，从而得到 $a \bmod q_i^{e_i}$，$i=1,2,\cdots,k$。再根据中国剩余定理，可以求得 $a \bmod (n-1)$，即求得 $\log_\alpha\beta$。

根据上述讨论，只有在 $n-1$ 的素数因子是小素数时，才能有效地分解 $n-1$ 求得 $q_i^{e_i}$，这就是 Pohlig-Hellman 算法的适用范围。

例 11.4 设 $n=29$，$\alpha=2$，$\beta=18$。计算 $\log_\alpha\beta$。

由于 $n-1=28=2^2\cdot7^1$。令 $\log_\alpha\beta=a$，下面计算 $a \bmod 2^2=a_0+a_1\cdot2^1$。首先计算列表 L：

$$\alpha^{0\cdot(n-1)/2} \bmod 29=2^0 \bmod 29=1$$
$$\alpha^{1\cdot(n-1)/2} \bmod 29=2^{28/2} \bmod 29=28$$

因为 $\beta^{(n-1)/2} \bmod 29=18^{28/2} \bmod 29=28$，故 $a_0=1$。令

$$\beta_1=\beta\alpha^{-1} \bmod 29=18\cdot2^{-1} \bmod 29=9$$

因为：

$$\beta_1^{n-1}/2^2 \bmod 29=9^{28/4} \bmod 29=28$$

故 $a_1=1$。因此，$a \bmod 2^2=a_0+a_1\cdot2^1=3$。

下面计算 $a \bmod 7^1=a_0$，首先利用公式 $\alpha^{s(n-1)/7} \bmod 29$，$0\leqslant s\leqslant6$。

计算得到 $1(s=0)$，$16(s=1)$，$24(s=2)$，$7(s=3)$，$25(s=4)$，$23(s=5)$，$20(s=6)$。因为 $\beta^{(n-1)/7} \bmod 29=18^{28/7} \bmod 29=25$，故 $a_0=4$。因此 $a \bmod 7^1=a_0=4$。最后，得到同余方程组：

$$a \equiv 3 \bmod 2^2$$
$$a \equiv 4 \bmod 7^1$$

根据中国剩余定理，可以求得 $a=11 \bmod 28$。因此，在 Z_{29} 中 $\log_2 18=11$。

算法 11.3　Pohig-Hellman(G,n,α,β,q,e)

```
/* 计算 a0,a1,…,ae-1,用于最终求 logα β mod n */
/* 输入：n-1的因子分解后的素数因子 q,及其幂次 e */
/* 输出：a0,a1,…,ae-1,即可用其求得 a mod q^e */
{
  j←0
  βj←β
  While j≤c-1
  {δ←βj^(n/q^(j+1))
  find i s.t. δ=a^(in/q)                    //s.t.使得
  aj←i
  βj+1←βj α^(-aj q^j)
  j←j+1}
  Return(a0,…,ac-1)
}
```

算法的时间复杂度是 $O(cq)$，经过观察，发现寻找满足 $\delta=\alpha^{in/q}$ 的值 i，可视为解一个特殊的离散对数问题，即 $i=\log_{\alpha^{n/q}}\delta$，元素 $\alpha^{n/q}$ 的阶是 q，所以每个 i 可用 $\sqrt{q}$ 时间计算，这样算法的复杂度可降为 $O(c\sqrt{q})$。

思　考　题

[1] 编写程序实现本章介绍的四种离散对数算法，并进行性能比较。
[2] 利用高性能计算机测试计算离散对数问题的四种算法。

第 12 章 其他高级应用*

本章最后给出一些高级应用，如 GM 公钥密码算法和 CRT 在秘密共享中的应用。本章重点是基于 CRT 的秘密共享，难度是 GM 加密算法。

12.1 平方剩余在 GM 加密中的应用*

针对确定性加密存在的问题，1982 年，S. Goldwasser 与 S. Micali 提出了概率公钥密码系统（probabilistic encryption scheme，也有文献称为随机化加密）的概念，简单解释就是加密体制对同一明文进行两次加密得到的密文有可能不同。并提出了一个概率公钥密码系统，称为 Goldwasser-Micali 公钥密码系统。概率公钥密码体制可达到更严格的安全目标——语义安全（Sematic Security）。

定义 12.1　多项式安全。如果一个被动敌手不能在期望的多项式时间内选择两个明文消息 m_1,m_2，且以大于 1/2 的概率正确区分 m_1,m_2 的加密结果。

定理 12.1　一个公钥加密方案是语义安全的，当且仅当它是多项式安全的。

Goldwasser-Micali 加密体制的安全性是基于平方剩余问题的困难性假设。平方剩余问题是指：如果不知道 n 的素数因子分解，那么要确定模 n 的平方剩余是困难的。

该体制是概率加密，即相同的明文，加密得到的密文是不同的。因为在加密的时候，引入了随机数，加密的密文和随机数有关（这一概率加密的思想也应用到 5.4 节 ElGamal 加密方案和 7.4 节 NTRU 加密方案中）。

Goldwasser-Micali 加密体制描述如下：

密钥生成：随机选定大素数 p 和 q，计算 $n=pq$。随机选定一个正整数 t 满足：$L(t,p)=L(t,q)=-1$，即 t 是模 p 和 q 的平方非剩余。(n,t)是公钥，p 和 q 是私钥。这里 $L(t,p)$与 $L(t,q)$均表示 Legendre 符号。

加密过程：设要加密的明文的二进制表示为 $m=m_1m_2\cdots m_s$。对每个明文比特 m_i 随机选择整数 x_i，$1\leqslant x_i\leqslant n-1$，计算：

$$c_i\equiv\begin{cases}tx_i^2 \bmod n, & m_i=1\\ x_i^2 \bmod n, & m_i=0\end{cases}$$

得到密文 $c=(c_1,c_2,\cdots,c_s)$。

解密过程：待解密的密文 $c=(c_1,c_2,\cdots,c_s)$。对每个密文项 c_i，先计算出 $L(c_i,p)$ 以及 $L(c_i,q)$ 的值，然后令

$$m_i=\begin{cases}1, & L(c_i,p)=L(c_i,q)=-1\\0, & L(c_i,p)=L(c_i,q)=1\end{cases}$$

得到解密后的明文 $m=m_1m_2\cdots m_s$。

例 12.1 假设私钥是(5,7)，即 $p=5,q=7$，选择 $t=3$，且满足其为 p 和 q 的平方非剩余，即 $L(t,p)=L(t,q)=-1$，公钥为(35,3)，明文为 $m=(11010)$，求加密、解密过程。

解答：

加密方法为：$c_i\equiv\begin{cases}tx_i^2 \bmod n, & m_i=1\\x_i^2 \bmod n, & m_i=0\end{cases}$，明文为(11010)，$t=3$。

加密过程如下：

$$c_1\equiv3\cdot8^2 \bmod 35\equiv17 \bmod 35$$
$$c_2\equiv3\cdot4^2 \bmod 35\equiv13 \bmod 35$$
$$c_3\equiv3^2 \bmod 35\equiv9 \bmod 35$$
$$c_4\equiv3\cdot6^2 \bmod 35\equiv3 \bmod 35$$
$$c_5\equiv7^2 \bmod 35\equiv14 \bmod 35$$

得到的密文为(17,13,9,3,14)

下面讲解解密过程：

利用私钥(5,7)和解密算法 $m_i=\begin{cases}1, & L(c_i,p)=L(c_i,q)=-1\\0, & L(c_i,p)=L(c_i,q)=1\end{cases}$。

对密文(17,13,9,3,14)，进行解密：

$$L(c_1,p)=L(17,5)=17^{(5-1)/2} \bmod 5=-1$$
$$L(c_1,q)=L(17,7)=17^{(7-1)/2} \bmod 7=-1$$

故 $m_1=1$。同理可得 $m_2=m_4=1,m_3=m_5=0$，从而明文为(11010)。

讨论：

如果在选取随机数的时候，恰为 p 或者 q 的倍数，则在解密时得到的 Legendre 值必有一个为 0，这时解出的明文比特视为 1，即当且仅当 $L(c_i,p)=L(c_i,q)=1$ 时，返回 0，否则为 1。

效率：

Goldwasser-Micali 的加密方法的主要问题是：由于是逐比特加密，加密后数据扩展了 $\log_2 n$ 倍(从 1 个比特的明文，转变为 $\log_2 n$ 长的密文)，因此应用该密码体制进行加密时运算量大，速度慢，只适用于单个二进制比特的加密和解密。

安全性分析：

首先给出几个事实。

(1) 设 p 是一个奇素数，a 是 Z_p^* 的一个生成元，则 $a\in Z_p^*$ 为模 p 的平方剩余，当且仅当 $a=a^i \bmod p$，其中 i 是一个偶数，因此 $|Q_p|=(p-1)/2$，$|\overline{Q}_p|=(p-1)/2$。即 Z_p^* 中

平方剩余和平方非剩余各占一半。

(2) 设 n 为两个互不相同的素数 p 和 q 的乘积，则 $a\in Z_n^*$ 是模 n 的平方剩余，当且仅当 $a\in Q_p$，$a\in Q_q$（这里 Q_p，Q_q 分别表示模 p 和模 q 的平方剩余）。因此，$|Q_n|=|Q_p|\cdot|Q_q|=(p-1)(q-1)/4$，$|\bar{Q}_n|=3(p-1)(q-1)/4$。

(3) 设 $n\geqslant 3$ 为一个奇整数，并令 $J_n=\left\{a\in Z_n^* \mid \left(\frac{a}{n}\right)=1\right\}$。模 n 的伪平方集定义为 J_n-Q_n，以$\widetilde{Q_n}$表示。

(4) 设 $n=pq$ 为两个互不相同的奇素数的乘积，则 $|Q_n|=|\widetilde{Q_n}|=(p-1)(q-1)/4$，即 J_n 中一半元素为平方剩余，一半元素为伪平方。

由于 x 是从 Z_n^* 中随机选择的，$x^2 \bmod n$ 是模 n 的一个随机平方剩余，tx^2 是模 n 的一个随机伪平方。敌手截获者获取密文 c_i，计算雅克比符号（见 8.3 节定义 8.1）$\left(\frac{c_i}{n}\right)=1$。但是，不论 $m_i=0,1$，均有 $\left(\frac{c_i}{n}\right)=1$，所以敌手得不到关于明文的任何信息，只能猜测。因此，Goldwasser-Micali 方案是语义安全的。

12.2 CRT 在秘密共享中的应用

12.2.1 秘密共享的概念

首先思考两个场景：

(1) 某个银行有三位出纳，他们每天都要开启保险库。为防止每位出纳可能出现的监守自盗行为，银行规定至少有两位出纳在场才能开启保险库。

(2) 核按钮通常掌握在三方手上，总统、国防部长、国防部，只有当三方中的两方在场时，才可以是最终控制核按钮。

上述两个问题可以利用秘密共享(Secret Sharing)方案来实现。所谓秘密分享就是将一个密钥分成许多份额(Share，又叫影子 Shadow)，然后秘密地分配给一些有关人员，使得某些有关人员在同时拿出他们的份额后可以重建密钥，而另外一些有关人员在同时拿出他们的份额后不能重建密钥。

严格地说，秘密共享技术的基本要求是将秘密 k 分成 n 个份额 $s_1,s_2,\cdots,s_n$：

(1) 已知任意 t 个 s_i 易于求出 k；

(2) 已知任意 $t-1$ 个 s_i 或更少的 s_i，不能确定 k。

因此，这种秘密共享也称为(t,n)门限(threshold)方案$(t\leqslant n)$。

秘密共享的概念是 Adi Shamir 于 1979 年给出的，Shamir 提出的方案是根据 Lagrange 插值公式构造了$(t\leqslant n)$门限方案，即为了重构 $t-1$ 次多项式，该多项式有 t 个未知的多项式系数，需要知道 t 个点，才能重构多项式曲线。如果只知道 $t-1$ 个点，则完全无法确定多项式系数。其示意图见图 12.1。

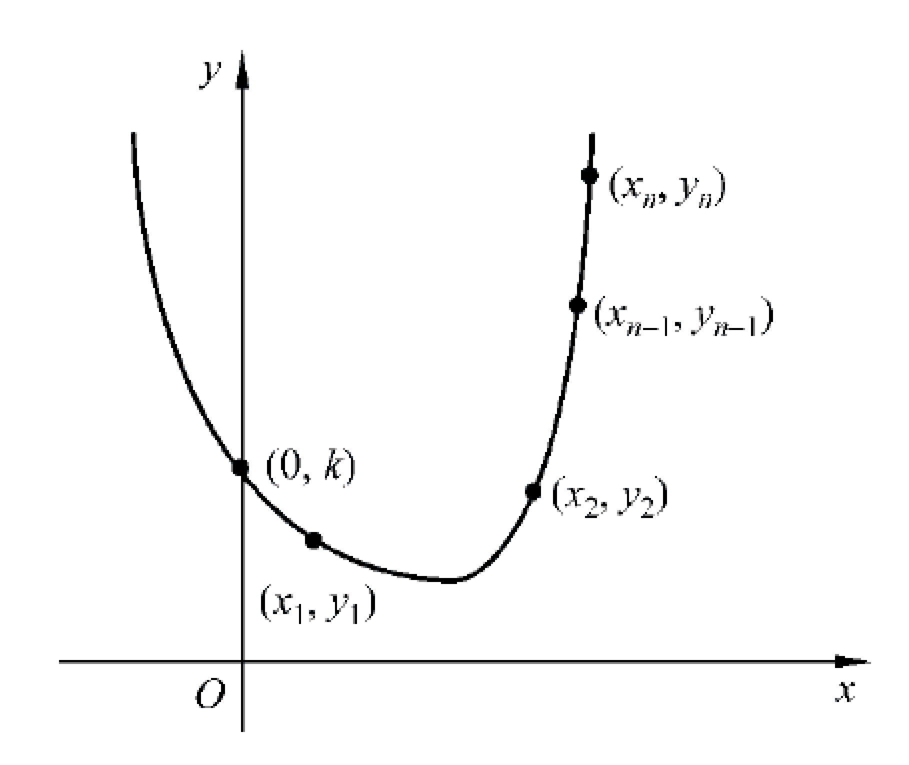


图 12.1　Adi Shamir 秘密共享方案

秘密共享技术提供了密钥抗泄露的可靠性，可以与其他密码学技术融合在一起，提供密码学技术的健壮性（提高抗共谋、容错、容入侵等能力）。秘密共享构成了门限密码学（Threshold Cryptography）的基础。

12.2.2　基于 CRT 的简单门限方案

基于中国剩余定理可以构造(t,n)门限方案，因为它有着类似于方程组的结构。

（1）参数设置：设 $m_1, m_2, \cdots, m_n$ 是 n 个严格递增的大于 1 的整数，且满足 $\gcd(m_i, m_j)=1(\forall i,j,i\neq j)$，以及 $m_n m_{n-1}\cdots m_{n-t+2}<m_1 m_2\cdots m_t$。这说明，$t-1$ 个 m_i 乘积的最大值，小于 t 个 m_i 乘积的最小值。分发的份额是秘密 k 对这 n 个不同模数的剩余。需要共享的秘密数据 k 满足 $m_n m_{n-1}\cdots m_{n-t+2}<k<m_1 m_2\cdots m_t$。因此，秘密至少需要 t 个方程才能确定。

（2）秘密分发：计算 $M=m_1 m_2\cdots m_n$，$s_i\equiv k \pmod{m_i}(i=1,2,\cdots,n)$。$(s_i, m_i, M)$是分发的份额。集合$\{(s_i,m_i,M)\}_{i=1}^{n}$即构成了一个$(t,n)$门限方案的份额集合。

（3）秘密重构：在 t 个参与者（记为 $i_1, i_2, \cdots, i_t$）中，每个 i_j 计算

$$\begin{cases} M_{i_j}=M/m_{i_j} \\ N_{i_j}=M_{i_j}^{-1} \pmod{m_{i_j}} \\ y_{i_j}=s_{i_j} M_{i_j} N_{i_j} \end{cases}$$

结合起来，根据中国剩余定理可求得

$$k=\sum_{j=1}^{t} y_{i_j}\left(\bmod \prod_{j=1}^{t} m_{i_j}\right)$$

显然，若参与者少于 t 个，则无法求出秘密 k。

例 12.2　设 $t=3, n=5, m_1=97, m_2=98, m_3=99, m_4=101, m_5=103$，秘密数据 $k=671875$，满足 $10403=m_4 m_5<k<m_1 m_2 m_3=941094$。

计算 $M=m_1 m_2 m_3 m_4 m_5=9790200882$，$s_i=k \pmod{m_i}(i=1,2,\cdots,5)$得 $s_1=53, s_2=85, s_3=61, s_4=23, s_5=6$。5 个份额为$(53, m_1=97, M=9790200882)$，$(85,98,9790200882)$，$(61,99,9790200882)$，$(23,101,9790200882)$，$(6,103,9790200882)$。

现在假定 i_1, i_2, i_3 联合起来计算 s，分别计算：

$$\begin{cases} M_1 = M/m_1 = 100929906 \\ N_1 = M_1^{-1} (\bmod\ m_1) = 95 \end{cases}$$
$$\begin{cases} M_2 = M/m_2 = 99900009 \\ N_2 = M_2^{-1} (\bmod\ m_2) = 13 \end{cases}$$
$$\begin{cases} M_3 = M/m_3 = 98890918 \\ N_3 = M_3^{-1} (\bmod\ m_3) = 31 \end{cases}$$

得到

$$\begin{aligned} k &= s_1 M_1 N_1 + s_2 M_2 N_2 + s_3 M_3 N_3 (\bmod\ m_1 m_2 m_3) \\ &= 53 \cdot 100929906 \cdot 95 + 85 \cdot 99900009 \cdot 13 \\ &\quad + 61 \cdot 98890918 \cdot 31 (\bmod\ 97 \cdot 98 \cdot 99) \\ &= 805574312593 (\bmod\ 941094) \\ &= 671875 \end{aligned}$$

可见秘密正确。现在假定 i_1, i_4, i_5 联合起来计算 s，分别计算：

$$\begin{cases} M_1 = M/m_1 = 100929906 \\ N_1 = M_1^{-1} (\bmod\ m_1) = 95 \end{cases}$$
$$\begin{cases} M_4 = M/m_4 = 96932682 \\ N_4 = M_4^{-1} (\bmod\ m_4) = 61 \end{cases}$$
$$\begin{cases} M_5 = M/m_5 = 95050494 \\ N_5 = M_5^{-1} (\bmod\ m_5) = 100 \end{cases}$$

得到

$$\begin{aligned} k &= s_1 M_1 N_1 + s_4 M_4 N_4 + s_5 M_5 N_5 (\bmod\ m_1 m_4 m_5) \\ &= 53 \cdot 100929906 \cdot 95 + 23 \cdot 96932682 \cdot 61 \\ &\quad + 6 \cdot 95050494 \cdot 100 (\bmod\ 97 \cdot 101 \cdot 103) \\ &= 70120825956\ (\bmod\ 1009091) \\ &= 671875 \end{aligned}$$

可见，秘密正确。现在假定 i_1, i_4 联合起来计算 k，分别计算：

$$\begin{aligned} k &= s_1 M_1 N_1 + s_4 M_4 N_4 (\bmod\ m_1 m_4) \\ &= 53 \cdot 100929906 \cdot 95 + 23 \cdot 96932682 \cdot 61 (\bmod\ 97 \cdot 101) \\ &= 644178629556 (\bmod\ 9797) \\ &= 5679 \end{aligned}$$

得到的秘密不正确。其原因是发生了“折回”，即 $671875 \bmod 9797 = 5679$。

容易发现这一方案的问题是区间 $[m_n m_{n-1} \cdots m_{n-t+2}, m_1 m_2 \cdots m_t]$ 不够大，容易猜测出 k。于是引出了下一节要介绍的方案 Asmuth-Bloom 秘密共享方案。

12.2.3 Asmuth-Bloom 秘密共享方案

有了上一节的铺垫，本节的内容就容易理解。将上一节的方案进行了推广，Asmuth 和 Bloom 于 1980 年提出了基于中国剩余定理 (t,n) 门限方案，同上，份额是由秘密 k 推算

出的数 y 对不同模数 $m_1, m_2, \cdots, m_n$ 的剩余。

(1) 参数设置。令 q 是一个大素数，$m_1, m_2, \cdots, m_n$ 是 n 个严格递增的数，且满足下列条件：

① $q>k$；

② $\gcd(m_i, m_j)=1, \forall i, j, i \neq j$；

③ $\gcd(q, m_i)=1, i=1,2,\cdots,n$；

④ $M=\prod_{i=1}^{t} m_i > q\prod_{j=1}^{t-1} m_{n-j+1}$。

条件①表明秘密 k 必须小于 q；条件②指出 n 个模数两两互素(可构成中国剩余定理的方程)；条件③表示 n 个模数都与 q 互素；条件④指出，$t-1$ 个模数 m_i 之积的最大值，即使乘上 q，也没有 t 个模数 m_i 之积的最小值 M 大。这是 Asmuth-Bloom 秘密共享方案对上一节方案的主要扩展之处。上一节的方案中 $t-1$ 个模数 m_i 之积最大值 $\mathrm{Max}(\prod^{t-1})$ 与 t 个模数 m_i 之积最小值 $\mathrm{Min}(\prod^{t})$之间至少可以容纳一个待共享的秘密，而扩展方案则可以容纳更多的待共享的秘密。

(2) 秘密分发。

首先，随机选取整数 A 满足 $0 \leqslant A \lfloor M/q \rfloor - 1$，并公布 q 和 A；

其次，$y=k+Aq$，则有 $y<q+Aq=(A+1)q \leqslant \lfloor M/q \rfloor \cdot q \leqslant M$；即秘密 k 放大了 Aq。

最后，计算 $y_i=y \bmod m_i (i=1,2,\cdots,n)$。$(m_i, y_i)$即为一个份额，将其分别传送给 n 个用户。

集合$\{(m_i, y_i) \mid i=1,2,\cdots,n\}$即构成了一个$(t,n)$门限方案。

(3) 秘密重构。

当 t 个参与者 $i_1, i_2, \cdots, i_t$ 提供出自己的子份额，由$\{(m_{i_j}, y_{i_j}) \mid i=1,2,\cdots,t\}$建立方程组

$$\begin{cases} y \equiv y_{i_1} \bmod m_{i_1} \\ y \equiv y_{i_2} \bmod m_{i_2} \\ \qquad \vdots \\ y \equiv y_{i_k} \bmod m_{i_k} \end{cases}$$

根据中国剩余定理可求得

$$y \equiv y' \bmod M'$$

其中，$M'=\prod_{j=1}^{t} m_{i_j} \geqslant M$。然后由 $y'-Aq$ 即得秘密 k。

正确性证明。

因为由 t 个成员的共享计算得到的模满足条件 $y<M \leqslant M'$，所以解出的 y 是唯一的，就是 $y' \bmod M'$。再由 $y'-Aq$ 即得秘密 k。

若仅有 $t-1$ 个参与者提供自己的份额(m_i, y_i)，条件是 $y'' \equiv y \bmod M''$，但只能求得 $y'' \equiv y \bmod M''$，式中 $M''=\prod_{j=1}^{k-1} m_{i_j}$，$y$ 发生了“折回”(即 $y>M''$)。由条件④得 $M''<M/q$，即 $M/M''>q$。

下面说明这种“折回”至少有 q 次。令 $y=y''+\alpha M''$，其中 $0\leqslant\alpha<\frac{y}{M''}<\frac{M}{M''}$，由于 $M/M''>q$，$(M'',q)=1$，当 α 在$[0,q]$之间变化时，$y''+\alpha M''$都是 y 的可能取值，共 $q+1$ 个，因此无法确定哪个是正确的 y。

例 12.3 设秘密 $k=4$，要求构建一个(3,5)门限方案。

(1) 参数设置。设选取素数 $q=7$，5 个模数分别为 $m_1=17$，$m_2=19$，$m_3=23$，$m_4=29$，$m_5=31$。容易验证模数满足前 3 个条件。

又因为 $M=m_1\cdot m_2\cdot m_3=17\cdot 19\cdot 23=7429>q\cdot m_4\cdot m_5=7\cdot 29\cdot 31=6293$，则第 4 个条件也满足。

(2) 秘密分发。在 $\left[0,\left\lfloor\frac{7429}{7}\right\rfloor-1\right]=[0,1060]$之间随机取 $A=117$，求得 $y=k+Aq=4+117\cdot 7=823$，然后计算

$$y_1\equiv y \bmod m_1\equiv 823 \bmod 17\equiv 7$$
$$y_2\equiv y \bmod m_2\equiv 823 \bmod 19\equiv 6$$
$$y_3\equiv y \bmod m_3\equiv 823 \bmod 23\equiv 18$$
$$y_4\equiv y \bmod m_4\equiv 823 \bmod 29\equiv 11$$
$$y_5\equiv y \bmod m_5\equiv 823 \bmod 31\equiv 17$$

{(17,7),(19,6),(23,18),(29,11),(31,17)}即构成一个(3,5)门限方案。

(3) 秘密重构。若第 1、3、5 个成员想恢复秘密，则他们提供自己的份额{(17,7),(23,18),(29,11)}，建立方程组：

$$\begin{cases} y\equiv 7 \bmod 17 \\ y\equiv 18 \bmod 23 \\ y\equiv 11 \bmod 29 \end{cases}$$

由此得：

$$M=17\cdot 23\cdot 29=11339$$
$$M_1=23\cdot 29=667 \qquad M_1^{-1}=13$$
$$M_2=17\cdot 29=493 \qquad M_2^{-1}=7$$
$$M_3=17\cdot 23=391 \qquad M_3^{-1}=27$$

由中国剩余定理得：

$$y\equiv(7\cdot 667\cdot 13+18\cdot 493\cdot 7+11\cdot 391\cdot 27) \bmod 11339\equiv 823$$

所以，恢复秘密为 $k=y-Aq=823-117\cdot 7=4$。

思　考　题

[1] 编写程序实现 Goldwasser-Micali 加密体制。

[2] 编写程序实现 Asmuth-Bloom 秘密共享机制。

参 考 文 献

[1] 陈恭亮. 简明信息安全数学基础. 北京：高等教育出版社，2011.
[2] 陈恭亮. 信息安全数学基础. 北京：清华大学出版社，2004.
[3] 陈恭亮，信息安全数学基础(第二版). 北京：清华大学出版社，2014.
[4] 覃中平，张焕国等，信息安全数学基础. 北京：清华大学出版社，2006.
[5] 裴定一，徐祥，信息安全数学基础. 北京：人民邮电出版社，2002.
[6] S. Y. Yan(颜松远)，计算数论(第 2 版). 杨思熳等译. 北京：清华大学出版社，2008.
[7] 颜松远. Computational Number Theory and Modern Cryptography(计算数论与现代密码学)，原出版社 Wiley. 北京：高等教育出版社，2013.
[8] Douglas R. Stinson. 密码学原理与实践. 冯登国译. 北京：电子工业出版社，2003.
[9] Alfred Menezes 等. 应用密码学手册. 胡磊译. 北京：电子工业出版社，2005.
[10] 谢敏. 信息安全数学基础. 西安：西安电子科技大学出版社，2006.
[11] 聂旭云，廖永建. 信息安全数学基础. 北京：科学出版社，2013.
[12] 贾春福，钟安鸣，赵源超. 信息安全数学基础. 北京：清华大学出版社，北京交通大学出版社，2010.
[13] 罗守山，陈萍，罗群，辛阳. 信息安全数学基础. 北京：国防工业出版社，2011.
[14] 李继国，余纯武，张福泰，马春光等. 信息安全数学基础. 武汉：武汉大学出版社，2006.
[15] 吴晓平，秦艳琳. 信息安全数学基础. 北京：国防工业出版社，2009.
[16] 董丽华，胡予濮，曾勇. 数论与有限域. 北京：机械工业出版社，2010.
[17] 陈鲁生等. 现代密码学. 北京：科学出版社，2002.
[18] 裴定一，祝跃飞. 算法数论. 北京：科学出版社，2002.
[19] 谷利泽等. 现代密码学教程. 北京：北京邮电大学出版社，2009.
[20] 林东岱. 代数学基础与有限域. 北京：高等教育出版社，2006.
[21] 潘承洞等. 初等数论. 北京：北京大学出版社，2003.
[22] 闵嗣鹤，严士健. 初等数论(第 3 版). 北京：高等教育出版社，2003.
[23] 阮传概，孙伟. 近世代数及其应用. 北京：北京邮电大学出版社，2002.
[24] 沈世镒. 近代密码学. 桂林：广西师范大学出版社，1998.
[25] 王小云，王明强，孟宪萌. 公钥密码学的数学基础. 北京：科学出版社，2013.
[26] 万哲先. 代数和编码(第三版). 北京：高等教育出版社，2007.
[27] 张焕国等. 椭圆曲线密码学导论. 北京：清华大学出版社，2003.
[28] 张焕国，刘玉珍. 密码学引论. 武汉：武汉大学出版社，2004.
[29] 潘承洞，潘承彪. 简明数论. 北京：北京大学出版社，1998.
[30] 柯召，孙琦. 数论讲义(第 2 版). 北京：高等教育出版社，2001.
[31] William Stallings. 密码编码学与网络安全——原理与实践(第 4 版). 孟庆树，王丽娜，傅建明等译. 北京：电子工业出版社，2007.
[32] Trappe W，Washington L C. 密码学与编码理论. 王金龙等译. 北京：人民邮电出版社，2008.
[33] Neal Koblitz. A Course in Number Theory and Cryptography. 2nd. Springer，GTM，1994.
[34] Neal Koblitz，Algebraic Aspects of Cryptography，Springer，1998.
[35] S. Y. Yan，Number Theory for Computing (2nd)，Springer，1998.